universität wien

DISSERTATION | DOCTORAL THESIS

Titel | Title

Integrating Multiple Imperfect Sources of Information in Natural Language Systems

verfasst von | submitted by

Andreas Joseph Stephan B.Sc. (TUM) B.Sc. (TUM) M.Sc. (TUM)

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Doktor der technischen Wissenschaften (Dr.techn.)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 786 880
Dissertationsgebiet lt. Studienblatt | Field of study as it appears on the student record sheet:	Informatik
Betreut von | Supervisor:	Univ.-Prof. Dr. -Ing. Benjamin Roth B.Sc. M.Sc.

Acknowledgements

Completing a PhD is an incredible journey, and I am deeply grateful to everyone who supported me along the way. This thesis would not have been possible without their guidance, collaboration, and encouragement.

First and foremost, I want to express my heartfelt gratitude to my supervisor, Prof. Dr. Benjamin Roth, for being an exceptional mentor and a constant source of support. Your insightful guidance, willingness to listen, and friendly approach made all the difference throughout my doctoral journey.
I am also deeply thankful to Prof. Dr. Claudia Plant for her valuable feedback during our collaborative projects and for keeping the larger research group running smoothly.
Next, I want to thank Prof. Dr. Annemarie Friedrich and Prof. Dr. Phang Vu, who agreed to review this thesis and serve as members of my committee.
I extend my gratitude to Prof. Dr. Hinrich Schütze and Prof. Dr. Barbara Plank for hosting me for a short research visit and letting me participate in their group's activities to expand my research horizons.
Beginning my PhD during the Covid-19 pandemic was challenging. Therefore, I am particularly thankful to Lukas for introducing us to Vienna and PhD life. I am also thankful for the support and for having formed a lasting friendship with Simon, Ylli, Timo, and Martin, who made this journey all the more enjoyable.
I am grateful to my collaborators Vasiliki, Anastasiia, Lena, and Pedro for their enthusiasm and teamwork. I am further thankful to my other labmates, Yuxi, Lukas, Loris, and Erion, for such a supportive environment and all the fun we had over the years. I owe a special thanks to my external collaborators, Dawei, Jan, and Matthias, for their expertise and support during our joint projects.
For their organizational and administrative support, I want to thank Tanja, Ewald, and our very own "Wiener Original" Nurcan.
I want to thank my mentor, Bernhard, for encouraging me to start a PhD and for being part of my future professional journey.
Lastly, I am extremely grateful to my parents Luzia and Manfred, my friends, and my partner, Selma, for their unconditional support and encouragement. Their belief in me made everything possible.

To all of you, thank you for being part of this incredible chapter of my life.

Abstract

Natural language systems focus on the analysis, understanding, and generation of text. Typical applications are classification tasks, such as sentiment analysis, or generative tasks, such as machine translation or question answering. As these tasks are motivated by real-world problems, usually an intuitive understanding of the task or relevant related information is available. Most of the time, such information does not come as ideal ground truth directly usable by natural language systems, but it exhibits *imperfections*. Imperfect information comes in different forms. An example is the usage of a sentiment lexicon to label the sentiment of movie reviews. It may lack sufficient vocabulary coverage and miss contextual information in the review. Another example is the output of another machine learning system trained on different but related data. This motivates us to conduct research in different settings where multiple imperfect sources of information are available.

This thesis presents and is based on four peer-reviewed studies and one study under review. The contributions are separated into three parts, each corresponding to a specific setting of using imperfect information.

In the first part, we work on a setting where we assume the availability of multiple so-called labeling functions, i.e., functions that automatically label data, for example, based on dictionaries or regular expressions. We derive two methods, WeaNF and SepLL to improve the performance on classification tasks. WeaNF is based on the idea of modeling individual labeling functions probabilistically, and SepLL is based on the idea of modeling the classification decision in the latent space.

In the second part, we focus on image clustering. Specifically, we conduct research on the usage of model-generated textual descriptions of images for image clustering. Such generated descriptions are incomplete, partially erroneous, and possibly focus on irrelevant information. We find that clustering the generated text often outperforms clustering the visual information. Further, we derive a method using generated text to obtain multiple alternative clusterings for the same dataset.

In the third part, we present research on a system of multiple, imperfect, large language models. We analyze so-called "LLM judges". In our setting, a LLM judge evaluates the responses of two other candidate language models which give answers to mathematical reasoning tasks.

In summary, the presented research focuses on different tasks and different settings where multiple sources of imperfect information are available. We derive methods and analyze the performance of systems based on imperfect information. Our research demonstrates the usefulness of using imperfect information.

Kurzfassung

Natürliche Sprachsysteme fokussieren sich auf die Analyse, das Verständnis und die Generierung von Text. Typische Anwendungen sind Klassifikationsaufgaben, wie beispielsweise die Sentimentanalyse, oder generative Aufgaben, wie die maschinelle Übersetzung oder die Fragebeantwortung. Da diese Aufgaben durch reale Probleme motiviert sind, ist üblicherweise ein intuitives Verständnis der Aufgabe oder relevanter verwandter Informationen vorhanden. Meistens liegt solch eine Information nicht als ideale, direkt von natürlichen Sprachsystemen nutzbare Ground Truth vor, sondern sie weist Unvollkommenheiten auf. Unvollkommene Information tritt in verschiedenen Formen auf. Ein Beispiel ist die Verwendung eines Sentimentlexikons zur Klassifikation des Sentiments von Filmkritiken. Es kann an unzureichender Vokabularabdeckung mangeln und kontextuelle Informationen in der Kritik übersehen. Ein weiteres Beispiel ist die Ausgabe eines anderen maschinellen Lernsystems, das auf unterschiedlichen, aber verwandten Daten trainiert wurde. Dies motiviert uns, Forschung in verschiedenen Szenarien durchzuführen, in denen mehrere unvollkommene Informationsquellen verfügbar sind.
Diese Arbeit präsentiert und basiert auf vier durch Peer-Review begutachteten Studien und einer Studie, die sich im Begutachtungsprozess befindet. Der wissenschaftliche Beitrag dieser Arbeit ist in drei Teile gegliedert, die jeweils einem spezifischen Szenario der Verwendung unvollkommener Information entsprechen.
Im ersten Teil arbeiten wir an einem Szenario, in der wir die Verfügbarkeit mehrerer sogenannter Labeling-Funktionen annehmen, d. h. Funktionen, die Daten automatisch kennzeichnen, beispielsweise basierend auf Wörterbüchern oder regulären Ausdrücken. Wir leiten zwei Methoden ab, WeaNF und SepLL, um die Leistung bei Klassifikationsaufgaben zu verbessern. WeaNF basiert auf der Idee, einzelne Labeling-Funktionen probabilistisch zu modellieren, und SepLL basiert auf der Idee, die Klassifikationsentscheidung im latenten Raum zu modellieren.
Im zweiten Teil konzentrieren wir uns auf das Clustering von Bildern. Insbesondere forschen wir an der Verwendung von modellgenerierten textuellen Beschreibungen von Bildern, um Bilder zu clustern. Solche generierten Beschreibungen sind unvollständig, teilweise fehlerhaft und konzentrieren sich möglicherweise auf irrelevante Informationen. Wir stellen fest, dass das Clustern der generierten Texte oft das Clustern der visuellen Informationen übertrifft. Weiterhin leiten wir eine Methode ab, die generierten Text verwendet, um mehrere alternative Clusterings für denselben Datensatz zu erhalten.
Im dritten Teil präsentieren wir Forschung zu einem System aus mehreren, unvollkommenen, großen Sprachmodellen. Wir analysieren sogenannte „LLM-Judes“. In unserer Umgebung bewertet ein LLM-Judge die Antworten von zwei anderen Kandidaten-Sprachmodellen, die Antworten auf mathematische Denkaufgaben geben.
Zusammenfassend konzentriert sich die präsentierte Forschung auf verschiedene Aufgaben

und verschiedene Szenarien, in denen mehrere Quellen unvollkommener Information verfügbar sind. Wir leiten Methoden ab und analysieren die Leistung von Systemen, die auf unvollkommenen Informationen beruhen. Unsere Forschung demonstriert die Nützlichkeit der Verwendung unvollkommener Information.

Bibliographic note

Four papers of this thesis have already been published, and one paper is currently under review. This dissertation is based on the following articles:

Paper A Andreas Stephan and Benjamin Roth. 2022. WeaNF: Weak Supervision with Normalizing Flows. In Proceedings of the 7th Workshop on Representation Learning for NLP, pages 269–279, Dublin, Ireland. Association for Computational Linguistics.

Paper B Andreas Stephan, Vasiliki Kougia, and Benjamin Roth. 2022. SepLL: Separating Latent Class Labels from Weak Supervision Noise. In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3918–3929, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Paper C Andreas Stephan, Lukas Miklautz, Kevin Sidak, Jan Philip Wahle, Bela Gipp, Claudia Plant, and Benjamin Roth. 2024. Text-Guided Image Clustering. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2960–2976, St. Julian's, Malta. Association for Computational Linguistics.

Paper D Andreas Stephan, Lukas Miklautz, Collin Leiber, Pedro Henrique Luz De Araujo, Dominik Répás, Claudia Plant, and Benjamin Roth. 2024. Text-Guided Alternative Image Clustering. In Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024), pages 177–190, Bangkok, Thailand. Association for Computational Linguistics.

Paper E Stephan, Andreas, Dawei Zhu, Matthias Aßenmacher, Xiaoyu Shen, and Benjamin Roth. From calculation to adjudication: Examining LLM Judges on Mathematical Reasoning Tasks. arXiv preprint arXiv:2409.04168 (2024).

Bibliographic note

The following papers were written during the doctorate and are not included in this thesis:

Paper F Anastasiia Sedova, Andreas Stephan, Marina Speranskaya, and Benjamin Roth. 2021. Knodle: Modular Weakly Supervised Learning with PyTorch. In Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021), pages 100–111, Online. Association for Computational Linguistics.

Paper G Andreas Baumann, Andreas Stephan, and Benjamin Roth. 2023. Seeing through the mess: evolutionary dynamics of lexical polysemy. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 8745–8762, Singapore. Association for Computational Linguistics.

Paper H Dawei Zhu, Xiaoyu Shen, Marius Mosbach, Andreas Stephan, and Dietrich Klakow. 2023. Weaker Than You Think: A Critical Look at Weakly Supervised Learning. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 14229–14253, Toronto, Canada. Association for Computational Linguistics.

Paper I Lena Zellinger, Andreas Stephan, and Benjamin Roth. 2024. Counterfactual Reasoning with Knowledge Graph Embeddings. In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2753–2772, St. Julian's, Malta. Association for Computational Linguistics.

Paper J Peter Dolog, Ylli Sadikaj, Yllka Velaj, Andreas Stephan, Benjamin Roth, and Claudia Plant. 2024. The Impact of Cluster Centroid and Text Review Embeddings on Recommendation Methods. In Companion Proceedings of the ACM Web Conference 2024 (WWW '24). Association for Computing Machinery, New York, NY, USA.

Paper K Matthias Aßenmacher, Andreas Stephan, Leonie Weissweiler, Erion Çano, Ingo Ziegler, Marwin Härttrich, Bernd Bischl, Benjamin Roth, Christian Heumann, and Hinrich Schütze. 2024. Collaborative Development of Modular Open Source Educational Resources for Natural Language Processing. In Proceedings of the Sixth Workshop on Teaching NLP, pages 43–53, Bangkok, Thailand. Association for Computational Linguistics.

Paper L Vasiliki Kougia, Anastasiia Sedova, Andreas Joseph Stephan, Klim Zaporojets, and Benjamin Roth. 2024. Analysing zero-shot temporal relation extraction on clinical notes using temporal consistency. In Proceedings of the 23rd Workshop on Biomedical Natural Language Processing, pages 72–84, Bangkok, Thailand. Association for Computational Linguistics.

Contents

Acknowledgements i

Abstract iii

Kurzfassung v

Bibliographic note vii

I. Synopsis 1

1. Introduction 3
- 1.1. Motivation . . . 3
- 1.2. Research Goal . . . 4
- 1.3. Outline . . . 5

2. Background and Related Work 7
- 2.1. Natural Language Processing . . . 7
- 2.2. Models and Representations . . . 8
- 2.3. Weak Supervision . . . 9
- 2.4. Evaluation . . . 11

3. Contributions 13
- 3.1. Multiple Rules . . . 14
  - 3.1.1. WeaNF: Weak Supervision with Normalizing Flows . . . 14
  - 3.1.2. SepLL: Separating Latent Class Labels from Weak Supervision Noise 15
- 3.2. Modality Transfer . . . 17
  - 3.2.1. Text-Guided Image Clustering . . . 17
  - 3.2.2. Text-Guided Alternative Image Clustering . . . 18
- 3.3. Large Language Model Capabilities . . . 20

4. Conclusion 21
- 4.1. Discussion . . . 21
- 4.2. Future Work . . . 23
- 4.3. Final Remarks . . . 24

Bibliography 25

II. Contributing Articles 35

5. WeaNF: Weak Supervision with Normalizing Flows 37

6. SepLL: Separating Latent Class Labels from Weak Supervision Noise 51

7. Text-Guided Image Clustering 65

8. Text-Guided Alternative Image Clustering 85

9. From Calculation to Adjudication: Examining LLM Judges on Mathematical Reasoning Tasks 101

Part I.

Synopsis

1. Introduction

Machine learning uses data to automatically derive algorithms that tackle tasks without explicitly designing algorithms or mathematical descriptions. Therefore, machine learning methods enable the (partial) automation of tasks and derivation of insights across a wide range of domains. In particular, natural language processing (NLP) recently made rapid progress, demonstrating significant advancements in solving language-based tasks.
A key challenge is the need to annotate data, typically using well-defined input and output pairs, to train machine learning models for a given task or to evaluate the performance of existing models. However, creating annotated datasets is labor-intensive, difficult, and expensive, presenting a bottleneck for many applications. The traditional machine learning pipeline of task definition, data labeling, model training, and model evaluation neglects that, in many cases, imperfect information is readily available.
The work presented in this thesis explores and analyzes methodologies where multiple sources of imperfect information are available. We introduce different methods that take multiple sources of imperfect information to improve the performance of machine learning systems. Further, we examine different situations and evaluate the performance of these methods on a variety of tasks.

1.1. Motivation

In many real-world applications, imperfect information is readily available. It may be reused from existing applications, databases, or machine learning systems. Alternatively, it might be available through abstract human knowledge, which can be formalized by regular expressions or, more generally, code (Ratner et al., 2016). If these information sources are properly understood and utilized, they offer the potential to partially mitigate or circumvent the limitations of traditional annotation approaches. By leveraging readily available resources, we can improve the efficiency and effectiveness of the creation and evaluation of NLP systems.
In the following, we describe examples that are related to the papers this thesis is based on.

Sentiment Analysis. Sentiment analysis aims to determine the sentiment of a text, e.g., by classifying a tweet into positive or negative. Imperfect information sources, such as smiley faces, hashtags, or shorthand expressions such as ":)", offer valuable cues for sentiment analysis. Depending on the number of keywords used, these cues might be sparse and they do not account for factors such as sarcasm, which need contextual information.

Image Clustering. Image clustering involves grouping similar images together based on their content. Imperfect information includes user-provided descriptions, tags, or automatically generated textual descriptions. Although additional textual descriptions may be noisy, incomplete, or ambiguous, they may help refine the obtained clustering by providing semantic information about the images.

LLMs-as-a-Judge of Mathematical Reasoning. Large language models may be employed as automatic evaluators in the realm of mathematical reasoning. For instance, an LLM can be used to evaluate the logical flow of arguments by other large language models and "judge" whether their answers are correct. Although the LLM's judgment may not be definitive or correct, it may provide valuable insights or guidance.

To use such information, it is essential to research the potential to improve the capabilities of current systems and to understand the issues and limitations such sources exhibit.

1.2. Research Goal

The primary goal of this research is to explore how multiple sources of imperfect information can be harnessed to optimize natural language systems. The field of NLP underwent significant changes throughout the period this research was conducted, opening up multiple directions in investigating such settings.
Initially, in 2021, transformer encoder models like BERT (Vaswani et al., 2017; Devlin et al., 2019) dominated the NLP landscape. Amongst others, they excelled in classification tasks. An intuitive way to provide imperfect labeling information is to provide functions that label data automatically based on, e.g., keywords. This paradigm is called data programming (Ratner et al., 2016). We aimed to create methods that generalize the imperfect information given by those functions to improve the performance on classification tasks. This led to the development of two methods, WeaNF (Stephan and Roth, 2022), an approach based on neural probabilistic modeling, and SepLL (Stephan et al., 2022), which makes predictions from the latent space.
As multimodal models like CLIP (Radford et al., 2021), connecting visual and textual information advanced, we wondered how to explore the potential of imperfect signals in multimodal settings. We set out to investigate whether model-generated image descriptions could enhance image clustering. Such descriptions may be incomplete, incorrect, or focus on parts of the image that are less useful for clustering a dataset. This led to text-guided image clustering (Stephan et al., 2024b), comparing image-based clustering and clustering based on generated text, and to alternative text-guided image clustering (Stephan et al., 2024a), an extension deriving multiple sensible clusterings of the same dataset.
With the rise of powerful generative text models such as GPT-3 (Brown et al., 2020) and ChatGPT (OpenAI, 2023) and their relevance, given their availability to the general public, we turned our attention to large language models (LLM). We investigate a system of

multiple LLMs. More specifically, in Stephan et al. (2024c) we investigate the effectiveness of using LLMs as judges, evaluating the correctness of two other candidate LLMs.
In summary, this research investigates diverse settings where multiple sources of imperfect information are available in various forms and methods for utilizing such information to enhance the quality and reliability of existing NLP systems. We focus on robust evaluation for assessing performance across specific tasks and situations. By addressing these challenges, this work aims to contribute to understanding and improving the potential of imperfect information.

1.3. Outline

The following describes the structure of the thesis. In Part 1, an overview of the performed research is given. Chapter 1 gives a motivation, defines the research problem and objectives, and gives an overview. Chapter 2 discusses background and related work, reviewing foundational concepts in machine learning and natural language processing. Chapter 3 summarizes the contributions of this research. Chapter 4 concludes the thesis by summarizing the key findings, discussing their implications, and outlining potential future directions for exploring the use of imperfect data sources in machine learning. Finally, Part 2 of this thesis encompasses the papers on which this thesis is based and details the contributions of the individual authors.

2. Background and Related Work

This chapter aims to establish the context for the research presented in this thesis in a comprehensive but compact manner. It introduces the broader field of NLP, names and introduces the types of models and representations used, and gives background on weak supervision and evaluation.

2.1. Natural Language Processing

Natural Language Processing is a subfield of artificial intelligence that aims to enable computers to comprehend, interpret, and generate human language. In recent years, NLP has experienced significant progress, mainly due to the increased availability of textual datasets and computational resources.

In its early days, NLP mostly focused on symbolic approaches to define language systems precisely. An early, well-known example is ELIZA (Weizenbaum, 1966), which relied on keywords and pre-defined rules to simulate a chatbot. Researchers went on to build complex ontologies to represent knowledge in machine-readable formats and theories such as rhetorical structure theory (Mann and Thompson, 1988) to represent language in a format useful for natural language systems.

While such rule-based representations are interpretable, the real world is often too complex to develop a comprehensive set of rules. Researchers turned to statistical natural language processing (Manning and Schütze, 1999) to automatically understand the underlying statistical patterns. A typical modeling approach includes N-gram features to represent text and a graphical model, e.g., Hidden Markov models (Rabiner, 1989), to learn from data.

Nowadays, deep learning is the dominant approach in NLP (Goodfellow et al., 2016). By defining a goal in the form of a mathematical function, named a loss function, it automatically optimizes a model from data. During training, the loss is optimized based on input and output pairs, and during prediction, the model has to predict the output, given the input. An early example in NLP is the neural probabilistic language model (Bengio et al., 2003), which trains a neural network to perform language modeling, i.e., predicting the next word in a sequence. In Mikolov et al. (2013), the authors present a method to learn word embeddings, i.e., dense vector representations of words that capture semantic and syntactic relationships. This method uses neural networks to learn embeddings that are optimized to predict the context of a word in a sequence.

In NLP research, a model is commonly analyzed by evaluating how well it solves a particular task. These tasks include classifying text into predefined categories (e.g., sentiment analysis, topic classification), clustering similar texts together (e.g., document clustering), generating new text (e.g., machine translation, text summarization), extracting structured information from unstructured text (e.g., named entity recognition), or answering questions posed in natural language. Nowadays, large language models are general-purpose models and are typically tested on a variety of tasks to evaluate their generality and robustness.

2.2. Models and Representations

Now, we establish the necessary background on the used models and representations. Specifically, we focus on different types of the Transformer architecture.

Transformers. The Transformer architecture (Vaswani et al., 2017) represents a pivotal advancement in NLP. Transformers are deep learning models that leverage the self-attention mechanism to process sequential data. They possess the capability to process the entire input sequence in parallel. This design has led to significant gains in efficiency and paved the way for scaling up model size and training data, which resulted in massive performance increases. Transformers are broadly categorized into encoder-only models, decoder-only models, and encoder-decoder hybrids. In this thesis, we mainly make use of encoder and decoder models. Transformer-based models are state-of-the-art in many modalities, such as text (Vaswani et al., 2017), images (Dosovitskiy et al., 2021), video (Arnab et al., 2021), audio (Baevski et al., 2020a) or graphs (Yun et al., 2019).

Encoder Transformers. Encoder transformer models are bi-directional, i.e., they can process the entire input sequence in parallel. They are trained using masked language modeling (MLM), where random tokens in the input are masked, changed, or kept the same, and the model is trained to predict the original token using a bidirectional context. These models are *pre-trained* on large text corpora using MLM to learn general language representations. Subsequently, they are *fine-tuned* on specific downstream tasks using supervised learning with task-specific labeled data. Here, a multi-layer perceptron on top of a special classification token is trained to fine-tune such models on classification tasks. This approach was introduced in BERT (Devlin et al., 2019), and a more robustly optimized version was introduced in RoBERTa (Liu et al., 2020).

Decoder Transformers. Decoder transformer models are unidirectional, processing the input sequence sequentially. They predict the next token given all previous tokens. Representative models are the Generative Pretrained Transformer (GPT) (Radford et al., 2018), and its successors GPT-2 (Radford et al., 2019), GPT-3 (Brown et al., 2020), GPT-4 (OpenAI et al., 2024). Modern large language models are decoder-only models that undergo several training phases. First, they are pre-trained on massive text corpora

of several trillion tokens with the goal of predicting the next token given all previous tokens. This is called language modeling. Then, during the instruction tuning phase, they are fine-tuned on datasets of instructions and desired responses to better follow natural language instructions. Finally, many models incorporate Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017), where human preferences are used to further train the model to be more helpful and aligned with human values. These models can be used in zero-shot settings by describing tasks in natural language or in few-shot settings by providing example input-output pairs in context.

Text Representations. Traditional text representations include N-grams, i.e., sequences of n words, and Term Frequency-Inverse Document Frequency (TF-IDF) (Manning and Schütze, 1999). Modern representations include sentence transformers (Reimers and Gurevych, 2019a), which train text representations such that texts of similar meaning are close in the embedding space and texts with different meanings are far apart.

Multimodality. Transformers were also adapted to train models for different modalities, such as spoken language (Baevski et al., 2020b) or images (Dosovitskiy et al., 2021). The integration of textual and visual modalities has led to models such as CLIP (Radford et al., 2021) to represent images and text in the same vector space, and Image-to-Text systems that employ vision transformers, called ViTs, (Dosovitskiy et al., 2021) to encode images and transformer decoders to generate text. These models bridge the gap between vision and language, enabling tasks like image captioning and visual question answering (Antol et al., 2015a).

2.3. Weak Supervision

This thesis addresses the situation where multiple imperfect sources of information are available. In the first part of this thesis, we address this problem via the formalization of so-called labeling functions (Ratner et al., 2016). These are functions $\lambda : X \to Y \cup \emptyset$ that label data programmatically, where X is the input space, Y is the set of possible labels, and $\emptyset$ is the case where the labeling function abstains from labeling.

Example. Assume the task is to predict the sentiment of a movie review. Let X represent the set of all possible movie reviews, and let Y represent the set of possible sentiments, e.g., "positive" and "negative". A labeling function λ is designed to return a positive label for reviews that contain the word "happy" and a negative label for reviews that contain the word "sad". If neither word is present, the labeling function λ abstains from labeling.

There are various types of labeling functions using different types of available information. They can be created via dictionary matches, regular expressions, or other heuristics. It is also possible to create labeling functions via database queries, readily available APIs, or machine learning models. They exhibit multiple properties of imperfection:

1. **Noise**: Labels may be erroneous or inconsistent with ground truth. This arises from the heuristic nature of labeling functions and the inherent ambiguity in natural language.

2. **Support**: Labeling functions may not cover all data points or aspects of the data. In the example above, sentences expressing sentiment without using the words "happy" or "sad" remain unlabeled.

3. **Bias**: Labeling functions may exhibit systematic biases due to the specific heuristics employed. For instance, a function based on the presence of positive or negative emotions might be biased toward informal language styles.

4. **Overlap**: Different labeling functions may provide overlapping or redundant information, potentially leading to an over-emphasis on certain features or hinting at different class labels.

The weak supervision field uses standardized comprehensive benchmarks, such as Knodle (Sedova et al., 2021) and WRENCH (Zhang et al., 2021), to study the usage of labeling functions as learning signals. These benchmarks consist of a diverse range of datasets, tasks, and a distinct set of labeling functions. They are used to evaluate the performance of dedicated weak supervision models.

One line of research aims to leverage the information in m labeling function matches $Z \in \{0,1\}^{n \times m}$, where Z_{ij} means that labeling function j assigns a label to sample i, and the relationship of labeling functions and classes to generate a prediction y. Such methods are called label models and are often based on ideas from statistics (Ratner et al., 2016; Varma et al., 2019).

More recent efforts aim to integrate deep learning with weak supervision. An intuitive approach is to directly train on the labels predicted by label models. On the other hand, there are specifically designed, so-called end-to-end models, such as DENOISE (Ren et al., 2020) or COSINE (Yu et al., 2021), which make use of the prediction probabilities during the training process.

A related area of study is learning based on noisy labels (Song et al., 2022), which models noise in the training labels, often making explicit assumptions on the noise distribution, such as, e.g., symmetry of noise. Approaches include the usage of dedicated loss functions (Natarajan et al., 2013) or dedicated architectures, e.g., by modeling the noise explicitly in a latent space (Sukhbaatar et al., 2015; Goldberger and Ben-Reuven, 2017). However, weak supervision distinguishes itself by focusing on the programmatic creation of noisy labels through labeling functions rather than assuming inherent noise in the data.

We introduce two methods for learning based on labeling functions. WeaNF (Stephan and Roth, 2022) focuses on explicitly modeling labeling function matches using the generative model normalizing flows, and SepLL (Stephan et al., 2022) models the prediction in a latent space while being trained to predict the labeling function matches.

2.4. Evaluation

It is essential to evaluate the performance of machine learning and NLP models to assess their efficacy and pinpoint areas for improvement. Notably, evaluating these systems has emerged as a dedicated field of research (Chang et al., 2024), tackling unique challenges and problems. As the current generation of models grows increasingly powerful and general, designing robust, comprehensive, and meaningful evaluation schemes becomes more complex and more critical. This thesis contains two types of evaluation schemes:

1. Task-specific metrics which use *well-defined human annotated input-output pairs* to measure how well the system output matches the annotated ground-truth output.
2. *Machine learning systems* that *automatically* evaluate the output of another system with or without the availability of annotated ground-truth output.

In the contributed articles, we mostly use well-defined input/output pairs and the following tasks:

Classification. We evaluate classification tasks using accuracy, precision, recall, and F_1-score (Manning et al., 2008). Precision describes the fraction of relevant instances among all retrieved instances, where typically, in binary classification, relevant instances have a ground truth label of "1" and retrieved instances have a predicted label of "1". Recall describes the fraction of retrieved samples among all relevant samples. The F_1-score is the harmonic mean of precision and recall.

Clustering. To evaluate clustering tasks, typically classification datasets are used. The data is clustered and the similarity between clustering and ground truth labels is computed using normalized mutual information (NMI), adjusted rand index (ARI) (Vinh et al., 2010), and the silhouette score (Rousseeuw, 1987) to assess cluster quality. For instance, the NMI is defined as follows:

$$\mathrm{NMI}(U, V) = \frac{2 \cdot I(U; V)}{H(U) + H(V)} \tag{2.1}$$

where $I(U; V)$ is the mutual information between the random variable U, which corresponds to the predicted cluster assignments, and ground truth cluster assignments V.The entropies of predicted cluster assignments and ground truth labels are denoted by $H(U)$ and $H(V)$, respectively. The NMI is a metric to measure how much information is shared between the clustering and the ground truth labels.

LLM-as-a-Judge. Furthermore, this thesis explores the feasibility of employing LLMs-as-a-Judge (Zheng et al., 2023) to evaluate the performance of LLMs on mathematical reasoning tasks. Here, two different LLMs generate a solution for a given problem. Subsequently, a third LLM, a "judge" LLM, is then tasked to decide which outputs are correct. Unlike human annotation, the LLM-as-a-judge paradigm is scalable and enables

the evaluation of large datasets without reliance on costly human annotators. Despite these benefits, the approach is challenging because of potential reliability issues, where judgments may lack consistency or accuracy. The approach's effectiveness varies across different tasks or domains.

3. Contributions

The following chapter provides an overview of the research presented in this thesis. We outline the individual problems addressed, the motivation behind each study, and the corresponding contributions. We divide the body of research into three parts:

Multiple rules. We begin by assuming the availability of multiple human-engineered rules, provided as programming code automatically labeling data. We propose two methods, WeaNF (Stephan and Roth, 2022) and SepLL (Stephan et al., 2022), to leverage these rules as learning signals. These methods address challenges such as noisy rules, limited support, and contradicting rules by introducing probabilistic and latent-space modeling approaches to improve the performance on classification tasks given a set of rules.

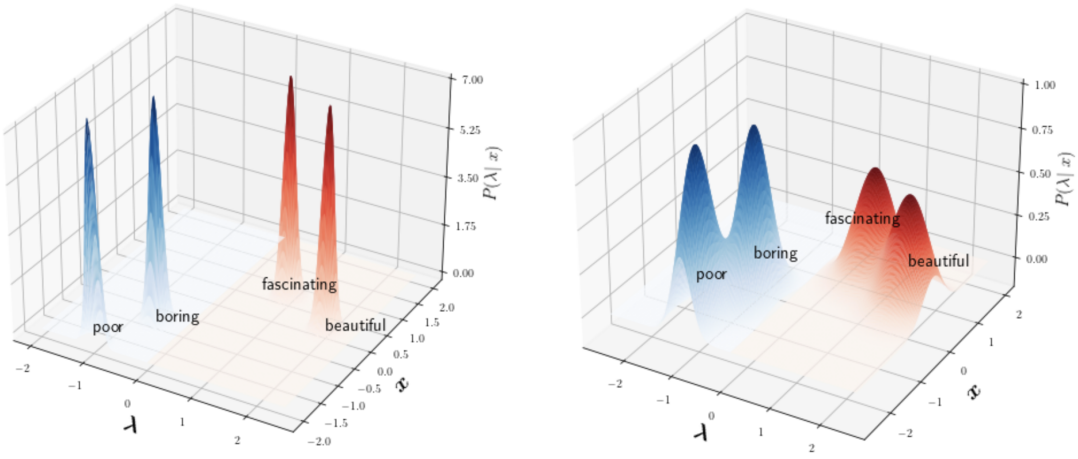
Multiple Modalities. We base this research on the motivation that it is possible to describe images using text and use text as a natural abstraction to describe useful clusters. In this research, we investigate the usage of model-generated textual descriptions of images. Such descriptions are incomplete (as we can not describe all pixels) and may be noisy (partially wrong). We introduce two methodologies, namely "text-guided image clustering" (Stephan et al., 2024b) and "alternative text-guided image clustering" (Stephan et al., 2024a), to cluster images based on the text generated by image-to-text models.

Multiple LLMs. Thirdly, we describe an analysis study (Stephan et al., 2024c) where we assume a system of three language models, where two language models derive answers to a mathematical reasoning problem, and a third language model, a "LLM judge" evaluates their answer. We analyze the properties related to imperfections in LLMs, as measured by performance.

3.1. Multiple Rules

In this line of research, we assume the availability of multiple rules, such as lookup dictionaries or regular expressions, that automatically assign labels to data points. This field is called weak supervision (Ratner et al., 2016). Naturally, these rules are imperfect and exhibit properties such as noise, incomplete coverage, or bias. As these rules are typically defined using code, we follow Ratner et al. (2016) and refer to these as labeling functions. We contribute two learning methods, based on the intuition of probabilistic modeling and latent space modeling, which use imperfect rules to train models to improve the performance on given tasks.

3.1.1. WeaNF: Weak Supervision with Normalizing Flows



(a) Subfigure 1 (b) Subfigure 2

Figure 3.1.: Overview of the WeaNF model. This example shows the task sentiment analysis and uses keywords to label data. Blue represents a negative sentiment, and red represents a positive sentiment. The X-axis represents the embedding of a rule λ, e.g., "poor" as an example of a keyword indicating a negative label. The Y-axis represents the embedding (or latent space) of a text input x. The Z-axis represents the learned density $P(\lambda|x)$ of a rule λ matching a text sample x.

In Stephan and Roth (2022), we introduce WeaNF, a model that uses probabilistic modeling to perform classification based on the described rules. The goal of this research was to explicitly model the probabilities of a single rule λ over text, using the generative model normalizing flows (Papamakarios et al., 2021), and subsequently use these probabilities to model the classification. This is based on the intuition that it is beneficial to access the probabilities of a rule matching a text sample, instead of using the rule to directly predict a label.

As illustrated in Figure 3.1, WeaNF first embeds both input text x, using a transformer encoder, and labeling functions λ, using a trained embedding, into latent spaces. These embeddings are then concatenated to form a joint representation. The model employs the generative process of normalizing flows to learn the probability $P(\lambda|x)$ of a rule matching a given text sample. We test multiple schemes to aggregate multiple rule probabilities $P(\lambda_i|x), i \in \{1, \ldots, n\}$ into a final classification probability.

We demonstrate the effectiveness of WeaNF on various commonly used weak supervision datasets, showing that weakly supervised normalizing flows compare favorably to standard weak supervision baselines. It is further possible to analyze the relationship between the rule-matching probability and the classification probability.

3.1.2. SepLL: Separating Latent Class Labels from Weak Supervision Noise

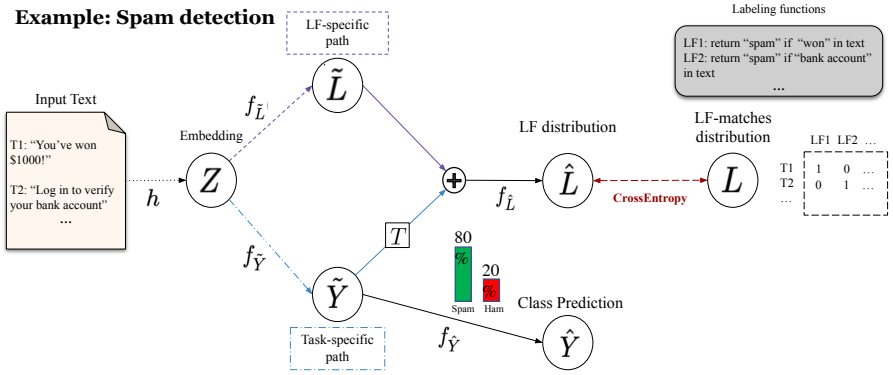


Figure 3.2.: Overview of the SepLL model. Text gets embedded into a latent space Z using a Transformer encoder, then this representation is transformed into a labeling function (LF) specific (or rule-specific) representation and a task-specific representation using multi-layer perceptrons. The task-specific representation is translated back into the LF space and combined using addition. A cross-entropy loss optimizes whether the combined distribution matches the true rule (LF) distribution. The classification is made from the latent layer, without any direct supervision.

In Stephan et al. (2022), we present SepLL, a method that aims to separate two types of complementary information associated with labeling functions: information related to the target label (task-specific information) and information specific to each labeling function (labeling function-specific information). During training, there is only access to the labeling function matching information, so the motivation is to train a model that learns to separate both types of information, using the information that is present in the data, the labeling function matching information, and the information that is present in

the pre-trained transformer.

SepLL introduces a branched deep model that splits labeling function information in the latent space (see Figure 3.2), enabling the model to learn directly from the rule-matching information. The intuition is that, during learning, the model separates task information from labeling function information by learning when two or multiple rules related to the same label match the same sample and relating that overlap to the text understanding capabilities present in the pre-trained transformer. After training, the task prediction is made from the latent space without any direct task signal during training.

On the WRENCH (Zhang et al., 2021) text classification benchmark, SepLL achieves state-of-the-art performance. While research on labeling function-based weak supervision stagnated in the subsequent years due to the rapid development of generative LLMs, SepLL is still a SOTA model in this specific setting of weak supervision. In summary, this research shows that building dedicated models for the information present in rule-based labeling functions is helpful to improve the performance on classification tasks.

3.2. Modality Transfer

Textual descriptions can serve as an abstraction, helping to navigate and interpret visual data effectively, even when the information provided by the textual description is incomplete. In this research, we build on advancements in image-to-text models (Li et al., 2023) to explore the usage of clusterings based on generated textual descriptions of images, in contrast to clusterings purely based on image features. These generated descriptions are imperfect because they can be partially incorrect and because they are incomplete, as not all pixels are described.

3.2.1. Text-Guided Image Clustering

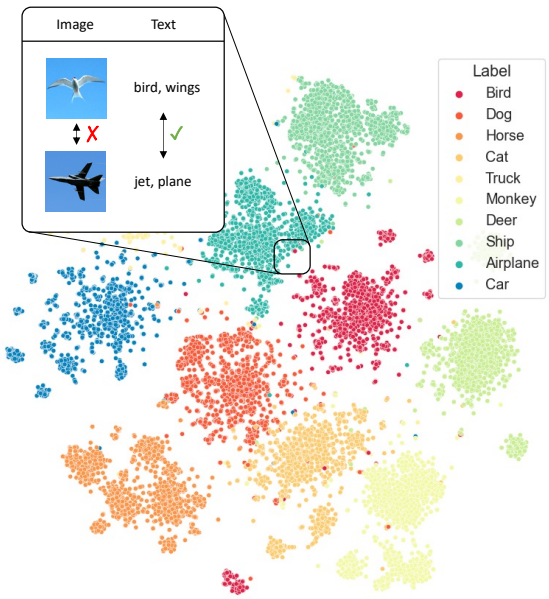


Figure 3.3.: The motivation of using text to cluster images. A t-SNE visualization of the BLIP-2 (Li et al., 2023) image embeddings for the STL10 dataset (Coates et al., 2011). While the images are highly similar on the pixel level (blue background and similar shape), the words bird and jet clearly distinguish the objects (and clusters).

In Stephan et al. (2024b), we introduce "Text-Guided Image Clustering", a methodology to generate textual descriptions for images using image-to-text models and then cluster these descriptions directly.

We experiment with image captioning (Vinyals et al., 2015) and visual question-answering models (Antol et al., 2015b) to obtain textual representations of images. The latter allows us to ingest domain knowledge into the clustering process, e.g., by prompting "What sport

is shown in the image?", which is often available in real-world applications. The generated text is often incomplete and misses details that image-to-text models cannot effectively capture, and is sometimes incorrect because of model biases and errors. Subsequently, we use text TF-IDF and sentence embeddings (Reimers and Gurevych, 2019b) to obtain text embeddings, which we cluster using K-Means.

Our experiments demonstrate that clustering representations based on model-generated text mostly outperforms clustering representations purely on image features. This suggests that language, even if imperfect, provides a useful abstraction that enhances clustering quality. Further, we introduce a word counting-based cluster explainability method and demonstrate that it is possible to generate interpretable textual cluster descriptions which provide a good idea of the examined data. For example, in one case, we found that all cluster descriptions match the ground truth cluster names, while the cluster accuracy is only 75%. In summary, the research shows how imperfect, generated textual descriptions of images can be utilized to obtain better image clusters.

3.2.2. Text-Guided Alternative Image Clustering

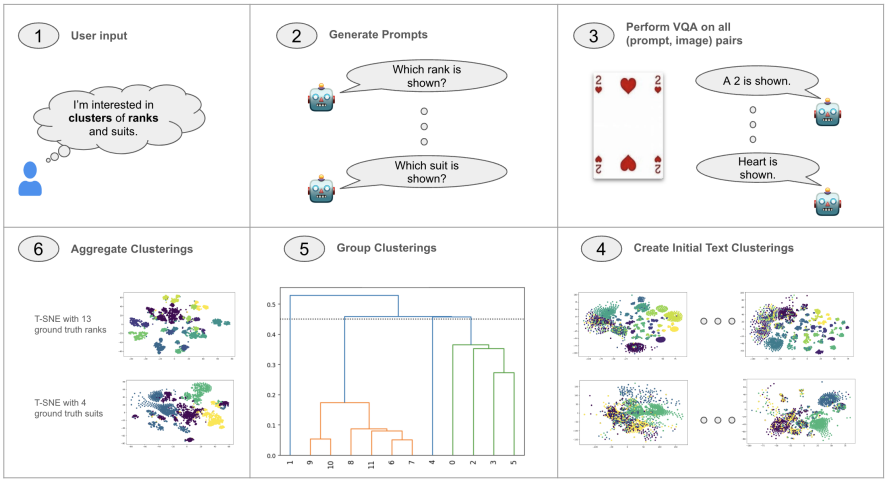


Figure 3.4.: An overview of the method. 1) A user provides interests. 2) An LLM generates a set of prompts to extract information from images. 3) A Vision-LLM generates answers. 4) For each prompt, the generated text is clustered (here, colors resemble rank and suit ground truth). 5) A hierarchy of similar clusterings is built. Based on a similarity threshold, clusterings are grouped (indicated by green and orange). 6) Groups are aggregated to obtain final alternative clusterings.

Image clustering is typically tested on rather simple classification datasets that have a unique ground-truth clustering. However, in real-world settings, it is often possible and sensible to obtain alternative clusterings corresponding to different characteristics. In

"Alternative Text-Guided Image Clustering" (Stephan et al., 2024a), we build on Stephan et al. (2024b) to address the need for diverse clusterings, using generated textual image descriptions.

An overview of the method is shown in 3.4. By leveraging LLMs, we automatically create diverse prompts, each emphasizing different facets. We use those prompts to generate multiple descriptions of the same image using VQA models. Note that this inherently introduces further noise and leaves gaps due to the limitations of the models. We then aggregate these varied perspectives using consensus clustering (Strehl and Ghosh, 2002) to derive robust, alternative clusterings that cater to different user queries.

Even though the generated text is noisy and incomplete, consensus-based aggregation of these imperfect descriptions can reveal new, meaningful groupings, often outperforming alternative clustering methods on top of image-only features.

3.3. Large Language Model Capabilities

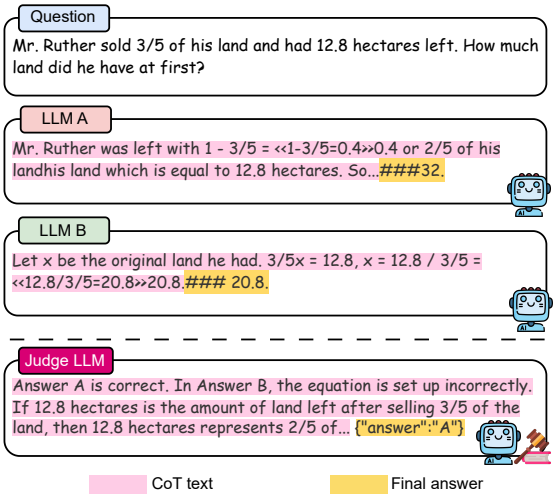


Figure 3.5.: Description of the problem setup. Two LLMs, A and B, provide candidate answers for a mathematical reasoning problem, and a judge LLM *has* to decide which answer, both answers, or no answer is correct. All LLMs use chain-of-thought reasoning (Wei et al., 2022) to generate a response.

Since the inception of GPT-3 (Brown et al., 2020) and ChatGPT (OpenAI, 2023), the capabilities and adoption of LLMs increased rapidly, implying the importance and need for research on such models. Therefore, connected to our overall research agenda, we investigate a system with multiple (imperfect) LLMs. The setup is shown in Figure 3.5. Two LLMs provide candidate answers to a mathematical reasoning problem (Hendrycks et al., 2021) and a third LLM, a "LLM judge" is tasked to decide which answer is correct or if both or no answer is incorrect. We call this 4-class classification the judgment task.

Our experiments show that LLM judges obtain a judgment classification performance of up to 80%. Further, we identify a correlation between task performance and judgment classification performance, suggesting that LLM judges are biased towards models of higher task accuracy. Our analysis contains usage recommendations where we find that LLM judges can consistently determine which candidate LLM has higher task performance. On the other hand, the experiments show that LLM judges can not be reliably used to improve performance on the given task. Rather, it is more beneficial to use all three LLMs (judge and candidate LLMs) to generate answers and subsequently perform a majority vote.

In summary, our results suggest that LLM judges can be useful in some settings, but practitioners should not blindly trust LLM judges and should carefully decide whether or not they should be used in their particular application.

4. Conclusion

In this thesis, we tackled multiple problems and settings with readily available imperfect sources of information.

First, we presented work on a specific weak supervision setup, where we assume the availability of functions that label data automatically using imperfect information such as keywords or regular expressions. We introduce two methods, WeaNF (Stephan and Roth, 2022) and SepLL (Stephan et al., 2022), which are trained on labeling function output to perform text classification.

Additionally, we explored the use of generated text to perform image clustering. We introduce two methods that address both standard (Stephan et al., 2024a) and alternative clustering scenarios (Stephan et al., 2024a). These approaches demonstrate the potential of integrating incomplete, potentially erroneous, or biased textual information to guide and improve image clustering.

Thirdly, we presented work on LLM judges (Stephan et al., 2024c) and investigated a setup of multiple (imperfect) LLMs where an LLM "judge" is tasked to evaluate the answer of two other candidate LLMs to a mathematical reasoning problem.

4.1. Discussion

A major theme in NLP research in the last years is the speed of development of the field. Therefore, the goal of this discussion is to touch on major points related to the presented research through the lens of "timelessness", i.e., topics that we think are likely to be relevant in the future, independent of the speed of development.

Availability of imperfect signals. In the context of this thesis, we presented research on three different settings involving imperfect information based on rules, modalities, and LLMs in Section 3. We argue that underlying ideas extend to many other domains or problems.
There exist many niche domains and data silos where organizations, companies, or groups keep their data private, rendering it difficult for models trained on publicly available data to understand such internal data which will impair their ability to solve tasks using such data. However, private information sources such as dictionaries, internal guidelines, customer chat logs, or documentation are often available. Such sources are often incomplete, contradict each other, or are not up-to-date, thus imperfect. Nevertheless, they can be

useful information in NLP systems, and be made accessible, e.g., via Retrieval Augmented Generation systems (Lewis et al., 2020).
In the context of NLP, in particular in the study of language models, many research areas are closely related. Research aims to generate synthetic data for pre-training (Gunasekar et al., 2023) and LLM post-training (Bai et al., 2022). This data is imperfect in the sense that it does not completely follow the complex distribution of real-world data. Another related strategy is distillation (Hinton et al., 2015) where one tries to use the output distribution of a (imperfect) teacher model, which is typically larger, to train a smaller student model. This technique has also been applied in the field of weak supervision to optimize a model iteratively (Yu et al., 2021).
While the capabilities of language models and machine learning models in general advance rapidly, we argue that it is worthwhile to analyze which sources of imperfect information are available and consider whether it is useful to incorporate them in the given problem or task.

Limited high-quality data vs. abundant imperfect data. Within the scope of this thesis, we built systems that employ imperfect information to build and improve machine learning systems. A directly related research question is whether it is better to use a limited amount of high-quality data or to use abundant, easier-to-obtain, imperfect data. Related to our research on weak supervision, the study by Zhu et al. (2023) showed that, for specific tasks, models fine-tuned on a rather small amount of clean labeled data may be sufficient to outperform models fine-tuned on weakly labeled, imperfect data. In contrast, Zhang et al. (2024) found on a different set of datasets that a substantial amount of clean labeled data is needed. In both studies, the authors find a "sweet spot" where the performance of models trained on high-quality data is better than models trained on imperfect data. Thus, it is always an individual decision based on how difficult it is to obtain high-quality data and how difficult it is to obtain imperfect data. Further, Zhu et al. (2023) observe in their experiments that using dedicated methods, it is best to combine both types of data.
This research question is relevant not only during fine-tuning but also during pre-training (Brown et al., 2020) or post-training (Bai et al., 2022). In the last years, LLMs were trained using increasingly large datasets, mostly crawled from the internet. Recent work, such as the Phi-series of language models (Gunasekar et al., 2023; Abdin et al., 2024), demonstrates that it is possible to train strong domain-specific models using an orders-of-magnitude smaller set of synthetic data. While it is unclear how this question evolves, it is also conceivable that a mixture of clean and imperfect data may be most useful.

Instance level vs. population level. A notable observation in our research is the distinction between outcomes at the instance level and the population level, i.e., of a subset of a dataset or a set of samples. In our research, methods based on imperfect signals demonstrate high performance at the population level, but their effectiveness varies at the instance level because of systematic and random biases.
This phenomenon was evident in our text-guided image clustering (Stephan et al., 2024a)

experiments. For example, even though the cluster descriptions, based on the generated texts, perfectly describe the ground truth cluster names, the actual clustering accuracy reached only approximately 75%. This discrepancy shows that imperfect information sources can provide valuable high-level insights suitable for exploratory data analysis, while potentially falling short in tasks requiring an exact grouping of samples.
Similarly, in Stephan et al. (2024c), we observed that the LLM judge is capable of determining which of two candidate LLMs has a higher performance on a given dataset. This is in contrast to the fact that they only obtain an 80% performance on the judgment classification task, i.e., determining which answers are correct or incorrect.
Here, clustering and LLM performance ranking serve as examples of settings where imperfect information enables concrete application on the population level, but less so on the instance level. We hypothesize that many other such use cases exist.

4.2. Future Work

In the following, we discuss future work in the context of this thesis. The following three paragraphs are each dedicated to a chapter in Section 3, each proposing research directions extending the work of a specific part of the contributions of this thesis.

Imperfection in language. In Stephan and Roth (2022) and Stephan et al. (2022), we have shown that rules, based on imperfect human intuitions, e.g., the keyword "good" is an indicator for a positive review, can be used to improve the performance of an NLP system. Generalizing this notion to language, we can define imperfect intuitions as an incomplete description of a task or an imperfect, "mostly correct", hint on how to solve a task. Future work may try to understand how to quantify levels of imperfection in language, how to quantify the impact of such imperfection on given models or systems, and how to leverage such imperfect intuitions to improve systems.

Modalities. Humans depend on multiple senses. The combination of multiple senses improves the overall system. If one sense is impaired, other senses can not fully compensate for its functionality. Similarly, a common hypothesis is that AI systems benefit from the integration of additional modalities. In Stephan et al. (2024b,a), we have shown that leveraging generated, imperfect, textual information is useful to tackle the image clustering task. Future work may apply a similar approach to audio, video, or graph data modalities. Other future research could systematically explore how a specific degree of imperfect signals, i.e., imperfect textual descriptions of images, as measured by coverage or incorrectness of information with respect to ground truth descriptions, impacts the performance of the system.

Multi-LLM. In Stephan et al. (2024c), we have analyzed a simple multi-LLM system where an LLM judge is used to evaluate the answer of two candidate LLMs. Our results show LLM judges are promising, but practitioners must be careful whether it makes sense to apply them to their specific use case. This represents a simple multi-LLM system.

With the increasing capabilities of LLMs, we expect LLMs to be increasingly able to communicate and, potentially, increase their ability to solve more difficult tasks. Such systems are always built on the assumption that all single LLMs are imperfect. Research in this direction could investigate communication protocols, consensus mechanisms, and their impact on performance and robustness.

4.3. Final Remarks

In this thesis, we investigated situations and problems where multiple imperfect sources of information are available. We made contributions to subfields of natural language processing and machine learning research. We have shown that there are relevant use cases and interesting research questions for imperfect data. While natural language processing and machine learning will continue to advance quickly, we think that there will always be a case for the usage of imperfect data.

Bibliography

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, Jyoti Aneja, Ahmed Awadallah, Hany Hassan Awadalla, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Caio César Teodoro Mendes, Weizhu Chen, Vishrav Chaudhary, Parul Chopra, Allie Del Giorno, Gustavo de Rosa, Matthew Dixon, Ronen Eldan, Dan Iter, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Jamie Huynh, Mojan Javaheripi, Xin Jin, Piero Kauffmann, Nikos Karampatziakis, Dongwoo Kim, Mahmoud Khademi, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Chen Liang, Weishung Liu, Xihui (Eric) Lin, Zeqi Lin, Piyush Madan, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Corby Rosset, Sambudha Roy, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Xia Song, Olatunji Ruwase, Xin Wang, Rachel Ward, Guanhua Wang, Philipp Witte, Michael Wyatt, Can Xu, Jiahang Xu, Weijian Xu, Sonali Yadav, Fan Yang, Ziyi Yang, Donghan Yu, Chengruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yunan Zhang, and Xiren Zhou. Phi-3 technical report: A highly capable language model locally on your phone. Technical Report MSR-TR-2024-12, Microsoft, August 2024. URL https://www.microsoft.com/en-us/research/publication/phi-3-technical-report-a-highly-capable-language-model-locally-on-your-phone/.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015a.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433, 2015b. doi: 10.1109/ICCV.2015.279.

Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6816–6826, 2021. doi: 10.1109/ICCV48922.2021.00676.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In H. Larochelle,

M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc., 2020a. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022. URL https://arxiv.org/abs/2212.08073.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3(null):1137–1155, March 2003. ISSN 1532-4435.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. A survey on evaluation of large language models. *ACM Trans. Intell. Syst. Technol.*, 15(3), March 2024. ISSN 2157-6904. doi: 10.1145/3641289. URL https://doi.org/10.1145/3641289.

Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/d5e2c0adad503c91f91df240d0cd4e49-Paper.pdf.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/coates11a.html.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=H12GRgcxg.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio Cesar, Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, Adam Tauman Kalai, Yin Tat Lee, and Yuanzhi Li. Textbooks are all you need, June 2023. URL https://www.microsoft.com/en-us/research/publication/textbooks-are-all-you-need/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. URL https://arxiv.org/abs/1503.02531.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/6b493230205f780e1bc26945df7481e5-Paper.pdf.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 19730–19742. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/li23q.html.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Ro{bert}a: A robustly optimized {bert} pretraining approach, 2020. URL https://openreview.net/forum?id=SyxS0T4tvS.

William C. Mann and Sandra A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text - Interdisciplinary Journal for the Study of Discourse*, 8(3):243–281, 1988. doi: doi:10.1515/text.1.1988.8.3.243. URL https://doi.org/10.1515/text.1.1988.8.3.243.

C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008. URL http://www-csli.stanford.edu/~hinrich/information-retrieval-book.html.

Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 1999. URL http://nlp.stanford.edu/fsnlp/.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*,

volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/3871bd64012152bfb53fdf04b401193f-Paper.pdf.

OpenAI. Chatgpt: Optimizing language models for dialogue. https://openai.com/blog/chatgpt/, 2023.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri

Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. URL http://jmlr.org/papers/v22/19-1028.html.

L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi: 10.1109/5.18626.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training, 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/6709e8d64a5f47269ed5cea9f625f7ab-Paper.pdf.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019a. URL https://arxiv.org/abs/1908.10084.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China, November 2019b. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. Denoising multi-source weak supervision for neural text classification. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3739–3754, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.334. URL https://aclanthology.org/2020.findings-emnlp.334/.

Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257.

Anastasiia Sedova, Andreas Stephan, Marina Speranskaya, and Benjamin Roth. Knodle: Modular weakly supervised learning with PyTorch. In Anna Rogers, Iacer Calixto, Ivan Vulić, Naomi Saphra, Nora Kassner, Oana-Maria Camburu, Trapit Bansal, and Vered Shwartz, editors, *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 100–111, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.repl4nlp-1.12. URL https://aclanthology.org/2021.repl4nlp-1.12/.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Andreas Stephan and Benjamin Roth. WeaNF":" weak supervision with normalizing flows. In Spandana Gella, He He, Bodhisattwa Prasad Majumder, Burcu Can, Eleonora Giunchiglia, Samuel Cahyawijaya, Sewon Min, Maximilian Mozes, Xiang Lorraine Li, Isabelle Augenstein, Anna Rogers, Kyunghyun Cho, Edward Grefenstette, Laura Rimell, and Chris Dyer, editors, *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 269–279, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.repl4nlp-1.27. URL https://aclanthology.org/2022.repl4nlp-1.27/.

Andreas Stephan, Vasiliki Kougia, and Benjamin Roth. SepLL: Separating latent class labels from weak supervision noise. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3918–3929, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.288. URL https://aclanthology.org/2022.findings-emnlp.288/.

Andreas Stephan, Lukas Miklautz, Collin Leiber, Pedro Henrique Luz De Araujo, Dominik Répás, Claudia Plant, and Benjamin Roth. Text-guided alternative image clustering. In Chen Zhao, Marius Mosbach, Pepa Atanasova, Seraphina Goldfarb-Tarrent, Peter Hase, Arian Hosseini, Maha Elbayad, Sandro Pezzelle, and Maximilian Mozes, editors, *Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024)*, pages 177–190, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. URL https://aclanthology.org/2024.repl4nlp-1.13/.

Andreas Stephan, Lukas Miklautz, Kevin Sidak, Jan Philip Wahle, Bela Gipp, Claudia Plant, and Benjamin Roth. Text-guided image clustering. In Yvette Graham and Matthew Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2960–2976, St. Julian's, Malta, March 2024b. Association for Computational Linguistics. URL https://aclanthology.org/2024.eacl-long.180/.

Andreas Stephan, Dawei Zhu, Matthias Aßenmacher, Xiaoyu Shen, and Benjamin Roth. From calculation to adjudication: Examining llm judges on mathematical reasoning tasks, 2024c. URL https://arxiv.org/abs/2409.04168.

Alexander Strehl and Joydeep Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 01 2002. doi: 10.1162/153244303321897735.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. January 2015. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Re. Learning dependency structures for weak supervision models. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6418–6427. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/varma19a.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(95):2837–2854, 2010. URL http://jmlr.org/papers/v11/vinh10a.html.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015. doi: 10.1109/CVPR.2015.7298935.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45, January 1966. ISSN 0001-0782. doi: 10.1145/365153.365168. URL https://doi.org/10.1145/365153.365168.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. Fine-tuning pre-trained language model with weak supervision: A contrastive-regularized self-training approach. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1063–1077, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.84. URL https://aclanthology.org/2021.naacl-main.84/.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/9d63484abb477c97640154d40595a3bb-Paper.pdf.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. WRENCH: A comprehensive benchmark for weak supervision. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://openreview.net/forum?id=Q9SKS5k8io.

Tianyi Zhang, Linrong Cai, Jeffrey Li, Nicholas Roberts, Neel Guha, and Frederic Sala. Stronger than you think: Benchmarking weak supervision on realistic tasks. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 122292–122315. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/pap

er_files/paper/2024/file/dd26d03d50af993ed052578c730e9729-Paper-Datasets_and_Benchmarks_Track.pdf.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=uccHPGDlao.

Dawei Zhu, Xiaoyu Shen, Marius Mosbach, Andreas Stephan, and Dietrich Klakow. Weaker than you think: A critical look at weakly supervised learning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14229–14253, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.796. URL https://aclanthology.org/2023.acl-long.796/.

Part II.

Contributing Articles

5. WeaNF: Weak Supervision with Normalizing Flows

Title	WeaNF: Weak Supervision with Normalizing Flows
Authors	Andreas Stephan and Benjamin Roth
Publication Outlet	In Proceedings of the 7th Workshop on Representation Learning for NLP, pages 269–279, Dublin, Ireland. Association for Computational Linguistics.
Copyright Information	
Thesis-Reference	Stephan and Roth (2022)

Supplementary Material

The code is available on Github at https://github.com/AndSt/wea_nf.

Author Contributions

Andreas Stephan	developed and expanded the initial idea. He formulated the probabilistic modeling and implemented the code. He designed the experiments, analyzed them, and iterated on them. Further, he wrote the majority of the paper.
Benjamin Roth	conceived the initial idea, provided feedback on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.

WeaNF: Weak Supervision with Normalizing Flows

Andreas Stephan
University of Vienna
andreas.stephan@univie.ac.at

Benjamin Roth
University of Vienna
benjamin.roth@univie.ac.at

Abstract

A popular approach to decrease the need for costly manual annotation of large data sets is weak supervision, which introduces problems of noisy labels, coverage and bias. Methods for overcoming these problems have either relied on discriminative models, trained with cost functions specific to weak supervision, and more recently, generative models, trying to model the output of the automatic annotation process. In this work, we explore a novel direction of generative modeling for weak supervision: Instead of modeling the output of the annotation process (the labeling function matches), we generatively model the input-side data distributions (the feature space) covered by labeling functions. Specifically, we estimate a density for each weak labeling source, or labeling function, by using normalizing flows. An integral part of our method is the flow-based modeling of multiple simultaneously matching labeling functions, and therefore phenomena such as labeling function overlap and correlations are captured. We analyze the effectiveness and modeling capabilities on various commonly used weak supervision data sets, and show that weakly supervised normalizing flows compare favorably to standard weak supervision baselines.

1 Introduction

Currently an important portion of research in natural language processing is devoted to the goal of reducing or getting rid of large labeled datasets. Recent examples include language model fine-tuning (Devlin et al., 2019), transfer learning (Zoph et al., 2016) or few-shot learning (Brown et al., 2020). Another common approach is weakly supervised learning. The idea is to make use of human intuitions or already acquired human knowledge to create weak labels. Examples of such sources are keyword lists, regular expressions, heuristics or independently existing curated data sources, e.g. a movie database if the task is concerned with TV shows. While the resulting labels are noisy, they provide a quick and easy way to create large labeled datasets. In the following, we use the term labeling functions, introduced in Ratner et al. (2017), to describe functions which create weak labels based on the notions above.

Throughout the weak supervision literature generative modeling ideas are found (Takamatsu et al., 2012; Alfonseca et al., 2012; Ratner et al., 2017). Probably the most popular example of a system using generative modeling in weak supervision is the data programming paradigm of Snorkel (Ratner et al., 2017). It uses correlations within labeling functions to learn a graph capturing dependencies between labeling functions and true labels.

However, such an approach does not directly model biases of weak supervision reflected in the feature space. In order to directly model the relevant aspects in the feature space of a weakly supervised dataset, we investigate the use of density estimation using normalizing flows. More specifically, in this work, we model probability distributions over the input space induced by *labeling functions*, and combine those distributions for better weakly supervised prediction.

We propose and examine four novel models for weakly supervised learning based on normalizing flows (**WeaNF-***): Firstly, we introduce a **standard** model **WeaNF-S**, where each labeling function is represented by a multivariate normal distribution, and its **iterative** variant **WeaNF-I**. Furthermore **WeaNF-N** additionally learns the **negative** space, i.e. a density for the space where the labeling function does not match, and a **mixed** model, **WeaNF-M**, where correlations of sets of labeling functions are represented by the normalizing flow. As a consequence, the classification task is a two step procedure. The first step estimates the densities, and the second step aggregates them to model label prediction. Multiple alternatives are discussed and analyzed.

Proceedings of the 7th Workshop on Representation Learning for NLP, pages 269 - 279

We benchmark our approach on several commonly used weak supervision datasets. The results highlight that our proposed generative approach is competitive with standard weak supervision methods. Additionally the results show that smart aggregation schemes prove beneficial.

In summary, our contributions are i) the development of multiple models based on normalizing flows for weak supervision combined with density aggregation schemes, ii) a quantitative and qualitative analysis highlighting opportunities and problems and iii) an implementation of the method[1]. To the best of our knowledge we are the first to use normalizing flows to generatively model labeling functions.

2 Background and Related Work

We split this analysis into a weak supervision and a normalizing flow section as we build upon these two areas.

Weak supervision. A fundamental problem in machine learning is the need for massive amounts of manually labeled data. Among others, weak supervision provides a way to counter the problem. The idea is to use human knowledge to produce noisy, so called weak labels. Typically, keywords, heuristics or knowledge from external data sources is used. The latter is called distant supervision (Craven and Kumlien, 1999; Mintz et al., 2009). In Ratner et al. (2017), data programming is introduced, a paradigm to create and work with weak supervision sources programmatically. The goal is to learn the relation between weak labels and the true unknown labels (Ratner et al., 2017; Varma et al., 2019; Bach et al., 2017; Chatterjee et al., 2019). In Ren et al. (2020) the authors use iterative modeling for weak supervision. Software packages such as SPEAR (**?**), WRENCH (**?**) and Knodle (Sedova et al., 2021) allow a modular use and comparison of weak supervision methods. A recent trend is to use additional information to support the learning process. Chatterjee et al. (2019) allow labeling functions to assign a score to the weak label. In Ratner et al. (2018) the human provided class balance is used. Additionally Awasthi et al. (2020); Karamanolakis et al. (2021) use semi-supervised methods for weak supervision, where the idea is to use a small amount of labeled data to steer the learning process.

[1] https://github.com/AndSt/wea_nf

Normalizing flows. While the concept of normalizing flows is much older, Rezende and Mohamed (2016) introduced the concept to deep learning. In comparison to other generative neural networks, such as Generative Adversarial networks (Goodfellow et al., 2014) or Variational Autoencoders (Kingma and Welling, 2014), normalizing flows provide a tractable way to model high-dimensional distributions. So far, normalizing received rather little attention in the natural language processing community. Still, Tran et al. (2019) and Ziegler and Rush (2019) applied them successfully to language modeling. An excellent overview over recent normalizing flow research is given in Papamakarios et al. (2021). Normalizing flows are based on the change of variable formula, which uses a bijective function $g : Z \to X$ to transform a base distribution Z into a target distribution X:

$$p_X(x) = p_Z(z) \left| \det \left(\frac{\partial g(z)}{\partial z^T} \right) \right|^{-1}$$

where Z is typically a simple distribution, e.g. multivariate normal distribution, and X is a complicated data generating distribution. Typically, a neural network learns a function $f : X \to Z$ by minimizing the KL-divergence between the data generating distribution and the simple base distribution. As described in Papamakarios et al. (2021) this is achieved by minimizing negative log likelihood

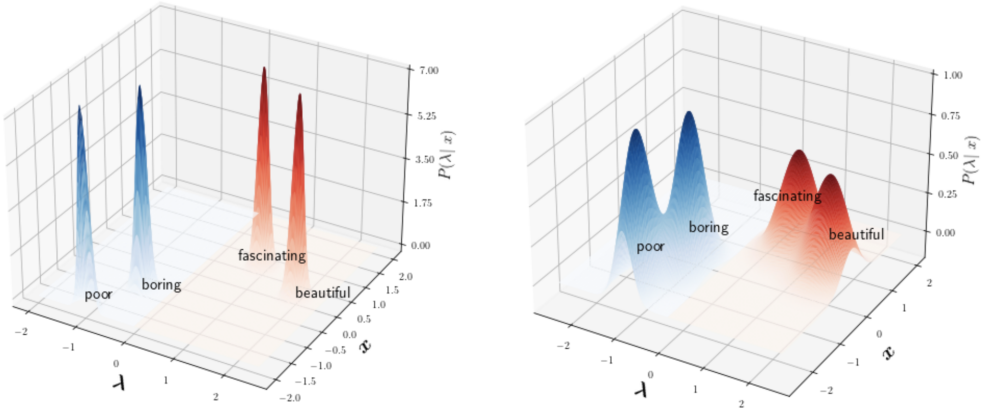
$$\log p_X(x) = \log p_Z(f(x)) + \log \left| \det \left(\frac{\partial f(x)}{\partial x^T} \right) \right|$$

The tricky part is to design efficient architectures which are invertible and provide an easy and efficient way to compute the determinant. The composition of bijective functions is again bijective which enables deep architectures $f = f_1 \circ \cdots \circ f_n$. Recent research focuses on the creation of more expressive transformation modules (Lu et al., 2021). In this work, we make use of an early, but well established model, called RealNVP (Dinh et al., 2017). In each layer, the input x is split in half and transformed according to

$$y_{1:d} = x_{1:d} \tag{1}$$

$$y_{d+1:D} = x_{d+1:D} \odot \exp\left(s(x_{1:d})\right) + t\left(x_{1:d}\right) \tag{2}$$

where $\odot$ is the pointwise multiplication and s and t neural networks. Using this formulation to realize



(a) Schematic view of the densities estimated by **WeaNF-S/I**. The concatenated input $[\boldsymbol{x}; \boldsymbol{\lambda}]$ is fed into the flow to learn the probability $P(x|\lambda)$. The graph shows the posterior $P(\lambda|x)$.

(b) **WeaNF-N** and **WeaNF-M** aim to smoothen the probability space, aiming to generalize more robustly to instances not directly matched by labeling functions.

Figure 1: Schematic overview of **WeaNF-***. The $X-$axis represents the labeling function embedding $\boldsymbol{\lambda}$, the $Y-$axis the text input $\boldsymbol{x}$. The $Z-$axis represents the learned density related to a labeling function. In this example we use the task sentiment analysis and keyword search as labeling functions. Blue denotes a negative sentiment and red a positive sentiment.

a layer f_i, it is easy and efficient to compute the inverse and the determinant.

Normalizing flows were used for semi-supervised classification (Izmailov et al., 2019; Atanov et al., 2020) but not for weakly supervised learning, which we introduce in the next chapter.

3 Model Description

In this section the models are introduced. The following example motivates the idea. Consider the sentence s, "The movie was fascinating, even though the graphics were poor, maybe due to a low budget.", the task sentiment analysis and labeling functions given by the keywords "fascinating" and "poor". Furthermore, "fascinating" is associated with the class POS, and "poor" with the class NEG. We aim to learn a neural network, which translates the complex object, text and a possible labeling function match, to a density, in the current example $P(s|\text{fascinating})$ and $P(s|\text{poor})$. We combine this information using basic probability calculus to make a classification prediction.

Multiple models are introduced. The standard model *WeaNF-S* naively learns to represent each labeling function as a multivariate normal distribution. In order to make use of unlabeled data, i.e. data where no labeling function matches, we iteratively apply the standard model *(WeaNF-I)*. Based on the observation that labeling functions overlap, we derive *WeaNF-N* modeling the negative space, i.e. the space where the labeling function does not match and the mixed model, *WeaNF-M*, using a common space for single labeling functions and the intersection of these. Furthermore, multiple aggregation schemes are used to combine the learned labeling function densities. See table 1 for an overview.

Before we dive into details, we introduce some notation. From the set of all possible inputs $\mathcal{X}$, e.g. texts, we denote an input sample by x and its corresponding vector representation by $\boldsymbol{x}$. The set of t labeling functions is $T = \{\lambda_1, \ldots, \lambda_t\}$ and the classes are $Y = \{y_1, \ldots, y_c\}$. Each labeling function $\lambda : \mathcal{X} \to \emptyset \cup \{y\}$ maps the input to a specific class $y \in Y$ or abstains from labeling. In some of our models, we also associate an embedding with each labeling function, which we denote by $\boldsymbol{\lambda} \in \mathbb{R}^h$. The set of labeling functions corresponding to label y is T_y.

WeaNF-S/I. The goal of the standard model is to learn a distribution $P(x|\lambda)$ for each labeling function λ. Similarly to Atanov et al. (2020) in semi-supervised learning, we use a randomly initialized embedding $\boldsymbol{\lambda} \in \mathbb{R}^h$ to create a representation for each labeling function in the input space. We concatenate input and labeling function vector and provide it as input to the normalizing flow, thus

	$P(y\|x) \propto$	WeaNF-S/I	WeaNF-N	WeaNF-M
Maximum	$\max_{\lambda \in T_y} P_\theta(x\|\lambda)$	√		√
Union	$\sum_{\lambda \in T_y} P(\lambda\|x)$		√	
NoisyOr	$1 - \prod_{\lambda \in T_y} (1 - P(\lambda\|x))$		√	
Simplex	$P\left(\left[\boldsymbol{x}; \frac{1}{\|T_y\|} \sum_{\lambda \in T_y} \boldsymbol{\lambda}\right]\right)$			√

Table 1: Overview over the used aggregation schemes. Note that $P(\lambda|x)$ is only accessible with WeaNF-N (see equation 4). Bold symbols denote vector representations.

learning $P([\boldsymbol{x}; \boldsymbol{\lambda}_i])$, where $[\cdot]$ describes the concatenation operation. A standard RealNVP (Dinh et al., 2017), as described in section 2 is used. See appendix B.1 for implementational details. In order to use the learned probabilities to perform label prediction, an aggregation scheme is needed. For the sake of simplicity, the model predicts the label corresponding to the labeling function with the highest likelihood, $y = \arg\max_{y \in Y} \max_{\lambda \in T_y} P(x|\lambda)$.

Additionally, to make use of the unlabeled data, i.e. the data points where no labeling function matches, an iterative version WeaNF-I is tested. For this, we use an EM-like (Dempster et al., 1977) iterative scheme where the predictions of the model trained in the previous iteration are used as labels for the unlabeled data. The corresponding pseudo-code is found in algorithm 1.

Algorithm 1 Iterative Model (WeaNF-I)

Require: $X_l \in \mathbb{R}^{n_l \times d}$, corresponding matches $\lambda_l \in \{0,1\}^{n_l \times t}$, unmatched $X_u \in \mathbb{R}^{n_u \times d}$
  F = train_flow(X_l, λ_l)
  for $i = 1, \ldots, r$ **do**
    $(\lambda_u)_i = \arg\max_\lambda F((X_u)_i; \lambda)$
    $X = \text{concat}(X_l, X_u)$, $\lambda = \text{concat}(\lambda_l, \lambda_u)$
    $F = \text{train\_flow}(X, \lambda)$
  end for

Negative Model. In typical classification scenarios it is enough to learn $P(x|y)$ to compute a posterior $P(y|x)$ by applying Bayes' formula twice, resulting in

$$P(y|x) = \frac{P(x|y)P(y)}{P(x|y)P(y) + P(x|\neg y)P(\neg y)} \quad (3)$$

where the class prior $P(y)$ is typically approximated on the training data or passed as a parameter. This is not possible in the current setting as often two labeling functions match simultaneously. In order to learn $P(\lambda|x)$, we explore a novel variant that additionally learns $P(x|\neg\lambda)$. The learning process is similar to $P(x|\lambda)$, so a second embedding $\tilde{\boldsymbol{\lambda}}$ is introduced to represent $\neg\lambda$. We optimize $P([\boldsymbol{x}; \boldsymbol{\lambda}]$ and $P\left(\left[\boldsymbol{x}; \tilde{\boldsymbol{\lambda}}\right]\right)$ simultaneously. In each batch I, the positive sample pairs $(x_i, \lambda_i)_{i \in I}$ and negative pairs (x_i, λ_j), sampled such that $(x_i, \lambda_j) \notin \{(x_i, \lambda_i)\}_{i \in I}$, are used to train the network. The number of negative samples per positive sample is an additional hyperparameter. Now Bayes' formula can be used as in equation 3 to obtain

$$P(\lambda|x) = \frac{P(x|\lambda)P(\lambda)}{P(x|\lambda)P(y) + P(x|\neg\lambda)P(\neg\lambda)}. \quad (4)$$

The access to the posterior probability $P(\lambda|x)$ provides additional opportunities to model $P(y|x)$. After initial experimentation we settled on two options. A simple addition of probabilities neglecting intersection probability, equation 5, which we call Union, and the NoisyOr formula, equation 7, which has previously shown to be effective in weakly supervised learning (Keith et al., 2017):

$$P(y|x) \propto \sum_{\lambda \in T_y} P(\lambda|x) \quad (5)$$

$$P(y|x) = P(\{\vee_{\lambda \in T_y} \lambda\}|x) \quad (6)$$

$$= 1 - \prod_{\lambda \in T_y} (1 - P(\lambda|x)) \quad (7)$$

Mixed Model. It was already mentioned that it is common that two or multiple labeling functions hit simultaneously. While WeaNF-N provides access to a posterior distribution which allows to model these interactions, the goal of the mixed model WeaNF-M is to model these intersections explicitly already in the density of the normalizing flow. More specifically, we aim to learn $P(x|\{\lambda_i\}_{i \in I})$ for arbitrary index families I. Once again, the embeddings space is used to achieve this

Dataset	#Classes	#Train / #Test samples	#LF's	Coverage(%)	Class Balance
IMDb	2	39741 / 4993	20	0.60	1:1
Spouse	2	8530 / 1187	9	0.30	1:5
YouTube	2	1440 / 229	10	1.66	1:1
SMS	2	4208 / 494	73	0.51	1:6
Trec	6	4903 / 500	68	1.73	1:13:14:14:9:10

Table 2: Some basic statistics describing the datasets. Coverage is computed on the train set by #matches / #samples.

goal. For a given sample x and a family I of matching labeling functions, we uniformly sample from the simplex of all possible combinations and obtain $\boldsymbol{\lambda}_I = \sum_{i \in I} \alpha_i \boldsymbol{\lambda}_i, \alpha_i \geq 0, \sum_{i \in I} \alpha_i = 1$. Afterwards we concatenate the weighted sum of the labeling function embeddings $\boldsymbol{\lambda}_I$ with the input x and learn $P([\boldsymbol{x}; \boldsymbol{\lambda}_I])$. Now that the density is able to access the intersections of labeling functions, we derive a new direct aggregation scheme. By σ_y we denote the simplex generated by the set of boundary points $\{\boldsymbol{\lambda}\}_{\lambda \in T_y}$. It is important to think about this simplex, as it theoretically describes the input space where the model learns the density related to class y. We use the naive but efficient variant which just computes the center of the simplex:

$$P(y|x) \propto P\left(\left[\boldsymbol{x}; \frac{1}{|T_y|} \sum_{\lambda \in T_y} \boldsymbol{\lambda}\right]\right) \tag{8}$$

Implementation. In practice, sampling of data points has to be handled on multiple occasions. Empirically and during the inspection of related implementations, e.g. the Github repository accompanying Atanov et al. (2020), we found that it is beneficial if every labeling function is seen equally often during training. It supports preventing a biased density towards specific labeling functions. When training WeaNF-N, the negative space is much larger than the actual space, so an additional hyperparameter controlling the amount of negative samples is needed. WeaNF-M aims to model intersecting probabilities directly. Most intersections occur too rarely to model a reasonable density. Thus we decided to only take co-occures into account which occur more often than a certain threshold. See appendix A.3 to get a feeling for the correlations in the used datasets.

4 Experiments

In order to analyze the proposed models experiments on multiple standard weakly supervised classification problems are performed. In the following, we introduce datasets, baselines and training details.

4.1 Datasets

Within our experiments, we use five classification tasks. Table 2 gives an overview over some key statistics. Note that these might differ slightly compared to other papers due to the removal of duplicates. For a more detailed overview of our preprocessing steps, see appendix A.1.

The first dataset is **IMDb** (Internet Movie Database) and the accompanying sentiment analysis task (Maas et al., 2011). The goal is to classify whether a movie review describes a positive or a negative sentiment. We use 10 positive and 10 negative keywords as labeling functions. See Appendix A.2 for a detailed description.

The second dataset is the **Spouse** dataset (Corney et al., 2016). The task is to classify whether a text holds a spouse relation, e.g. "Mary is married to Tom". Here, 90% of the samples belong to the no-relation class, so we use macro-F_1 score to evaluate the performance. As the third dataset another binary classification problem is given by the **YouTube** Spam (Alberto et al., 2015) dataset. The model has to decide whether a YouTube comment is spam or not. For both, the Spouse and the YouTube dataset, the labeling functions are provided by the Snorkel framework (Ratner et al., 2017).

The **SMS** Spam detection dataset (Almeida et al., 2011), we abbreviate by SMS, also asks for spam but in the private messaging domain. The dataset is quite skewed, so once again macro-F_1 score is used. Lastly, a multi-class dataset, namely **TREC-6** (Li and Roth, 2002), is used. The task is to classify questions into six categories, namely Abbreviation, Entity, Description, Human and Location. The labeling functions provided by (Awasthi et al., 2020) are used for the SMS and the TREC dataset. We

	IMDb	Spouse(F_1)	YouTube	SMS (F_1)	Trec
MV	56.84	49.87	81.66	56.1	61.2
MV + MLP	73.20	29.96	**92.58**	92.41	53.27
DP + MLP	67.79	57.05	88.79	84.40	43.00
WeaNF-S	73.06	52.28	89.08	86.71	67.4
WeaNF-I	**74.08**	**57.96**	89.08	**93.54**	**67.8**
WeaNF-N (NoisyOr)	72.96	54.60	90.83	79.63	54.8
WeaNF-N (Union)	71.98	50.83	91.70	83.48	60.2
WeaNF-M (Max)	70.16	55.16	85.15	88.23	49.8
WeaNF-M (Simplex)	63.53	56.91	86.03	76.29	25.4

Table 3: Comparison of baselines to our model variants. The numbers reflect accuracies, or F_1-scores, where explicitly mentioned. Names in parenthesis describe the aggregation mechanism.

took the preprocessed versions of the data available within the Knodle weak supervision programming framework (Sedova et al., 2021).

4.2 Baselines

Three baselines are used. While there are many weak supervision systems, most use additional knowledge to improve performance. Examples are class balance (Chatterjee et al., 2019), semi-supervised learning with very little labels (Awasthi et al., 2020; Karamanolakis et al., 2021) or multi-task learning (Ratner et al., 2018). To ensure a fair comparison, only baselines are used that solely take input data and labeling function matches into account. First we use majority voting (MV) which takes the label where the most rules match. For instances where multiple classes have an equal vote or where no labeling function matches, a random vote is taken. Secondly, a multi-layer perceptron (MLP) is trained on top of the labels provided by majority vote. The third baseline uses the data programming (DP) paradigm. More explicitly, we use the model introduced by Ratner et al. (2018) implemented in the Snorkel (Ratner et al., 2017) programming framework. It performs a two-step approach to learning. Firstly, a generative model is trained to learn the most likely correlation between labeling functions and unknown true labels. Secondly, a discriminative model uses the labels of the generative model to train a final model. The same MLP as for second baseline is used for the final model.

4.3 Training Details

Text input embeddings are created with the SentenceTransformers library (Reimers and Gurevych, 2019) using the *bert-base-nli-mean-tokens* model. They serve as input to the baselines and the normalizing flows. Hyperparameter search is performed via grid search over learning rates of $\{1e-5, 1e-4\}$, weight decay of $\{1e-2, 1e-3\}$ and epochs in $\{30, 50, 100, 300, 450\}$, and label embedding dimension in $10, 15, 20$ times the number of classes. Additionally, the number of layers is in $\{6, 8\}$, and the negative sampling value for WeaNF is in $\{2, 3\}$. The full set up ran 30 hours on a single GPU on a DGX 1 server.

5 Analysis

The analysis is divided into three parts. Firstly, a general discussion of the results is given. Secondly, an analysis of the densities predicted by WeaNF-N is shown and lastly, a qualitative analysis is performed.

5.1 Overall Findings

Table 3 exposes the main evaluation. The horizontal line separates the baselines from our models. For WeaNF-N and WeaNF-M, no iterative schemes were trained. This enables a direct comparison to the standard model WeaNF-I.

Interestingly, the combination of Snorkel and MLP's is often not performing competitively. In the IMDb data set there is barely any correlation between labeling functions, complicating Snorkel's approach. The large number of labeling functions e.g. Trec, SMS, could also complicate correlation based approaches. Appendix A.3 shows correlation graphs.

As indicated by the bold numbers, the WeaNF-I is the best performing model. Only on the YouTube dataset, an iterative scheme could not improve the results. Related to this observation, in Ren

Labeling Function	Example	Dataset	$P(x\|\lambda)$	Label (λ)	Gold	Prediction
won .* claim	...won ... call ...	SMS	↑	Spam	Spam	Spam
.* I'll .*	sorry, **I'll** call later	SMS	↑	No Spam	No Spam	No Spam
.* i .*	**i** just saw ron burgundy captaining a party boat so yeah	SMS	↓	No Spam	No Spam	No Spam
(explain|what) .* mean .*	**What** does the abbreviation SOS **mean** ?	Trec	↑	DESCR	ABBR	DESCR
(explain|what) .* mean .*	What are Quaaludes ?	Trec	↑	DESCR	DESCR	DESCR
who.*	**Who** was the first man to ... Pacific Ocean ?	Trec	↓	HUMAN	HUMAN	HUMAN
check .* out .*	**Check out** this video on YouTube:	YouTube	↑	Spam	Spam	Spam
#words < 5	subscribe my	YouTube	↑	Spam	Spam	No Spam
.* song .*	This **Song** will never get old	YouTube	↓	No Spam	No Spam	No Spam
.* dreadful .*	...horrible performance .... annoying	IMDb	↑	NEG	NEG	NEG
.* hilarious .*	...liked the movie...funny catchphrase...WORST...low grade...	IMDb	↑	POS	NEG	POS
.* disappointing .*	don't understand stereotype ... goofy ..	IMDb	↓	NEG	NEG	POS
.* (husband|wife) .*	...Jill.. she and her **husband**...	Spouse	↑	Spouses	Spouses	Spouses
.* married .*	... asked me to marry him and I said yes!	Spouse	↑	Spouses	No Spouses	Spouses
family word	Clearly excited, Coleen said: 'It's my eldest **son** Shane and Emma.	Spouse	↓	No Spouses	No Spouses	No Spouses

Table 4: Examples selected from the 10 most likely (↑) and 10 most unlikely (↓) combinations of sentences and labeling functions, using the density $P(x|\lambda)$ provided by WeaNF-I. Labeling function matches are bold. We observe that the flow often generalizes to unmatched examples. We slightly simplified some rules and shortened some texts in order to fit the page size.

	IMDb	**Spouse**	**YouTube**	**SMS**	**Trec**
Acc	72.38	74.04	78.17	88.71	72.63
P	5.93	5.1	38.95	23.3	13.65
R	37.53	39.31	55.01	44.34	61.07
F_1	10.25	9.02	45.61	30.55	22.31
Cov	4.31	5.74	19.31	3.01	4.39

Table 5: Evaluation of the labeling function prediction $P(\lambda|x)$. Precision, Recall and F_1 score are computed via the weighted average of the statistics of all labeling functions. Coverage is computed as #matches/#all possible matches.

Dataset	Labeling Fct.	Cov(%)	Prec	Recall
IMDb	*boring*	5.8	13.12	26.87
Spouse	family word	9. 0	16.53	35.96
YouTube	*song*	23.58	56.72	70.73
SMS	won *call*	0.81	66.67	1.0
Trec	how.*much	2.4	60.0	75.0

Table 6: Statistics for the labeling functions obtaining the highest F_1 score for the prediction $P(\lambda|x)$, using the WeaNF (NoisyOr) model.

Dataset	Labeling Fct.	Cov(%)	Prec	Recall
IMDb	*imaginative*	0.42	0.77	52.38
Spouse	spouse keyword	14.5	0	0
YouTube	person entity	2.62	6.45	33.33
SMS	I .* miss	0.6	0	0
Trec	what is .* name	2.2	2.26	100

Table 7: Same as table 6, but here the labeling functions obtaining the lowest F_1 score are shown. Only those are taken into account which occur more often than 10 times in the test set.

et al. (2020) the authors achieve promising results using iterative discriminative modeling for semi-supervised weak supervision.

WeaNF-N outperforms the standard model in three out of five datasets. We observe that these are the datasets with a large amount of labeling functions. Possibly, this biases the model towards a high value of $P(x|\neg\lambda)$ which confuses the prediction.

The simplex aggregation scheme only outperforms the maximum aggregation on two out of five datasets. We infer that the probability density over the labeling function input space is not smooth enough. Ideally, the simplex method should always have a high confidence in the prediction of a labeling function λ if its confident on the non-mixed embedding $\boldsymbol{\lambda}$ which is what Max is doing.

5.2 Density Analysis

We divide into a global analysis and a local, i.e. a per-labeling function, analysis. Table 5 pro-

vides some global statistics, table 6 and 7 subsequently show statistics related to the best and worst performing labeling function estimations. In the local analysis a labeling function is predicted if $P(\lambda|x) \geq 0.5$. The WeaNF-N model is used because it is the only model with direct access to $P(\lambda|x)$.

It is important to mention that in the local analysis, a perfect prediction of the matching labeling function is not wanted, as this would mean that there is no generalization. Thus, a low precision might be necessary for generalization, and a the recall would indicate how much of the original semantic or syntactic meaning of a labeling function is retained.

Interestingly, while the overall performance of WeaNF-N is competitive on the IMDb and the Spouse data sets, it is failing to predict the correct labeling function. One explanation might be that these are the data sets where the texts are substantially longer which might be complicated to model for normalizing flows. In table 7 typically the worst performing approximation of labeling function matches seems to be due to low coverage. An exception is the the Spouse labeling function.

5.3 Qualitative Analysis

In table 4 a number of examples are shown. We manually inspected samples with a very high or low density value. Note that density values related to $P(x|\lambda), \lambda \in T_y$ are functions f taking arbitrary values which only have to satisfy $\mathbb{E}_{x:\lambda(x)=y}[f(x)] = 1$.

We observed the phenomenon that either the same labeling functions take the highest density values $P(x|\lambda)$ or that a single sample often has a high likelihood for multiple labeling functions. In the table 4 one can find examples where the learned flows were able to generalize from the original labeling functions. For example, for the IMDb dataset, it detects the meaning "funny" even though the exact keyword is "hilarious".

6 Conclusion

This work explores the novel use of normalizing flows for weak supervision. The approach is divided into two logical steps. In the first step, normalizing flows are employed to learn a probability distribution over the input space related to a labeling function. Secondly, principles from basic probability calculus are used to aggregate the learned densities and make them usable for classification tasks. Motivated by aspects of weakly supervised learning, such as labeling function overlap or coverage, multiple models are derived each of which uses the information present in the latent space differently. We show competitive results on five weakly supervised classification tasks. Our analysis shows that the flow-based representations of labeling functions successfully generalize to samples otherwise not covered by labeling functions.

Acknowledgements

This research was funded by the WWTF through the project "Knowledge-infused Deep Learning for Natural Language Processing" (WWTF Vienna Research Group VRG19-008), and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - RO 5127/2-1.

References

T C Alberto, J V Lochter, and T A Almeida. 2015. TubeSpam: Comment Spam Filtering on YouTube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 138–143.

Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. 2012. Pattern Learning for Relation Extraction with a Hierarchical Topic Model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 54–59, Jeju Island, Korea. Association for Computational Linguistics.

Tiago A Almeida, J M G Hidalgo, and A Yamakami. 2011. Contributions to the study of SMS spam filtering: new collection and results. In *DocEng '11*.

Andrei Atanov, Alexandra Volokhova, Arsenii Ashukha, Ivan Sosnovik, and Dmitry Vetrov. 2020. Semi-Conditional Normalizing Flows for Semi-Supervised Learning.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from Rules Generalizing Labeled Exemplars.

Stephen H Bach, Bryan Dawei He, Alexander Ratner, and Christopher Ré. 2017. Learning the Structure of Generative Models without Labeled Data. *CoRR*, abs/1703.0.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen,

Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners.

Oishik Chatterjee, Ganesh Ramakrishnan, and Sunita Sarawagi. 2019. Data Programming using Continuous and Quality-Guided Labeling Functions. *CoRR*, abs/1911.0.

D Corney, M-Dyaa Albakour, Miguel Martinez-Alvarez, and Samir Moussa. 2016. What do a Million News Articles Look like? In *NewsIR@ECIR*.

Mark Craven and Johan Kumlien. 1999. Constructing Biological Knowledge Bases by Extracting Information from Text Sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86. AAAI Press.

A P Dempster, N M Laird, and D B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, 39(1):1–38.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2017. Density estimation using Real NVP.

Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks.

Pavel Izmailov, Polina Kirichenko, Marc Finzi, and Andrew Gordon Wilson. 2019. Semi-Supervised Learning with Normalizing Flows.

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan Awadallah. 2021. Self-Training with Weak Supervision.

Katherine Keith, Abram Handler, Michael Pinkham, Cara Magliozzi, Joshua McDuffie, and Brendan O'Connor. 2017. Identifying civilians killed by police with distantly supervised entity-event extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1547–1557, Copenhagen, Denmark. Association for Computational Linguistics.

Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes.

Xin Li and Dan Roth. 2002. Learning Question Classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Cheng Lu, Jianfei Chen, Chongxuan Li, Qiuhao Wang, and Jun Zhu. 2021. Implicit Normalizing Flows. In *International Conference on Learning Representations*.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. 2021. Normalizing Flows for Probabilistic Modeling and Inference.

Alexander Ratner, Stephen H Bach, Henry R Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid Training Data Creation with Weak Supervision. *CoRR*, abs/1711.1.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. 2018. Training Complex Models with Multi-Task Weak Supervision.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising Multi-Source Weak Supervision for Neural Text Classification. *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Danilo Jimenez Rezende and Shakir Mohamed. 2016. Variational Inference with Normalizing Flows.

Anastasiia Sedova, Andreas Stephan, Marina Speranskaya, and Benjamin Roth. 2021. Knodle: Modular Weakly Supervised Learning with PyTorch.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing Wrong Labels in Distant Supervision for Relation Extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 721–729, Jeju Island, Korea. Association for Computational Linguistics.

Dustin Tran, Keyon Vafa, Kumar Krishna Agrawal, Laurent Dinh, and Ben Poole. 2019. Discrete Flows: Invertible Generative Models of Discrete Data.

Paroma Varma, Frederic Sala, Ann He, Alexander Ratner, and Christopher Ré. 2019. Learning Dependency Structures for Weak Supervision Models.

positive	negative
beautiful	poor
pleasure	disappointing
recommendation	senseless
dazzling	second-rate
fascinating	silly
hilarious	boring
surprising	tiresome
interesting	uninteresting
imaginative	dreadful
original	outdated

Table 8: Keywords used to create rules for the IMDb dataset.

Zachary M Ziegler and Alexander M Rush. 2019. Latent Normalizing Flows for Discrete Sequences.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer Learning for Low-Resource Neural Machine Translation.

A Additional Data Description

A.1 Preprocessing

A few steps were performed, to create a unified data format. The crucial difference to other papers is that we removed duplicated samples. There were two cases. Either there were very little duplicates or the duplication occurred because of the programmatic data generation, thus not resembling the real data generating process. Most notably, in the spouse data set 60% of all data points are duplicates. Furthermore, we only used rules which occurred more often than a certain threshold as it is impossible to learn densities on only a handful of examples. The threshold is In order to have unbiased baselines, we ran the baseline experiments on the full set of rules and the reduced set of rules and took the best performing number.

A.2 IMDb rules

The labeling functions for the IMDb dataset are defined by keywords. We manually chose the keywords. We defined them in such a way that their meaning has rather little semantic overlap. The keywords are shown in table 8.

A.3 Labeling Function Correlations

In order to use labeling functions for weakly supervised learning, it is important to know the correlation of labeling functions to i) derive methods to combine them and ii) help to understand phenomena of the model predictions.

Thus we decided to add correlation plots. More specifically, we use the Pearson Correlation coefficient.

B Additional Implementationial Details

B.1 Architecture

As mentioned in section 3, the backbone of our flow is RealNVP architecture, which we introduced in section 2. With sticking to the notation in formula 2 the network layers to approximate the functions s and t are shown below

```
s = nn.Sequential(
    nn.Linear(dim, hidden_dim),
    nn.LeakyReLU(),
    nn.BatchNorm1d(hidden_dim),
    nn.Dropout(0.3),
    nn.Linear(hidden_dim, dim),
    nn.Tanh()
)
t = nn.Sequential(
    nn.Linear(dim, hidden_dim),
    nn.LeakyReLU(),
    nn.BatchNorm1d(hidden_dim),
    nn.Dropout(0.3),
    nn.Linear(hidden_dim, dim),
    nn.Tanh()
)
```

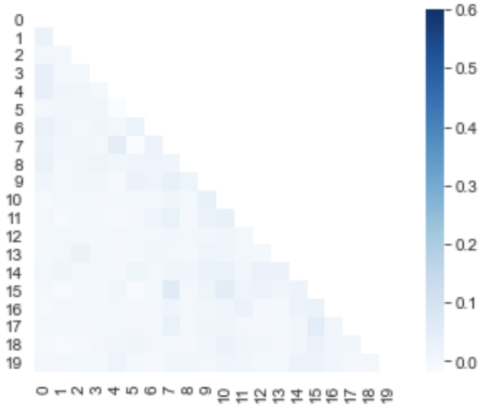
Hyperparameters are the depth, i.e. number of stacked layers, and the hidden dimension.

B.2 WeaNF-M Sampling

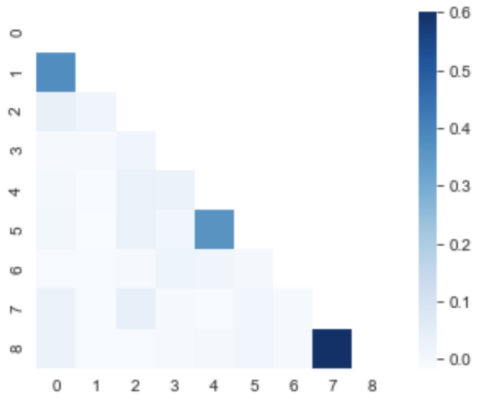
For the mixed model WeaNF-M the sampling process becomes rather complicated.

Next up, the code to produce the convex combination $\alpha_1, \ldots, \alpha_t$ is shown. The input tensor takes values in $\{0, 1\}$ and has shape $b \times t$ where b is the batch size and t the number of labeling functions.Note that some mass is put on every labeling functions. We realized that this bias imrpoves performance.

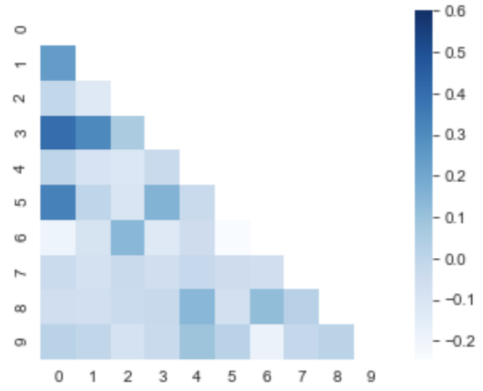
```
def weight_batch(self, batch_y: torch.Tensor):
    """Returns weighting array forming convex sum.
       Shape: (batch_dim, num_rules)
    """
    batch_y = batch_y.float()
    batch_y += 0.1 * torch.ones(batch_y.shape)
    batch_y = batch_y * torch.rand(batch_y.shape)
    row_sum = batch_y.sum(axis=1, keepdims=True)
    nbatch_y = batch_y / row_sum
    return nbatch_y
```



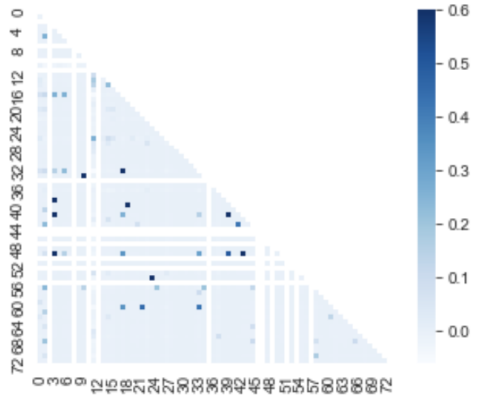
(a) IMDb



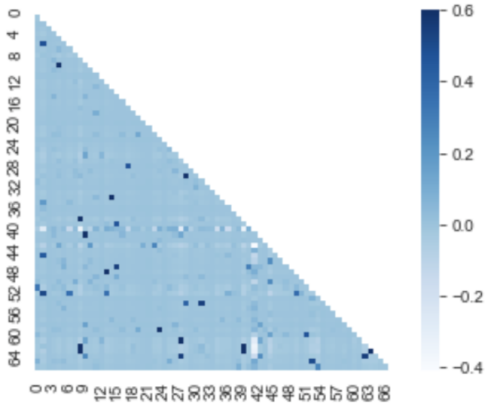
(b) Spouse



(c) YouTube



(d) SMS



(e) Trec

6. SepLL: Separating Latent Class Labels from Weak Supervision Noise

Title	SepLL: Separating Latent Class Labels from Weak Supervision Noise
Authors	Andreas Stephan, Vasiliki Kougia and Benjamin Roth
Publication Outlet	In Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3918–3929, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
Copyright Information	
Thesis-Reference	Stephan et al. (2022)

Supplementary Material

Code is available on Github at https://github.com/AndSt/sepll. Further, the model is integrated in the WRENCH (Zhang et al., 2021) benchmark on https://github.com/JieyuZ2/wrench.

Author Contributions

Andreas Stephan	conceived the initial idea, developed and expanded on the initial idea. He formulated the branched model and implemented the code. He designed the experiments, conducted the experiments, analyzed them, and iterated on them. Further, he wrote the majority of the paper.
Vasiliki Kougia	participated in the analysis of the experiments and helped write the paper.
Benjamin Roth	participated in developing and expanding the initial idea, provided feedback on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.

SepLL: Separating Latent Class Labels from Weak Supervision Noise

Andreas Stephan[1,2] **Vasiliki Kougia**[1,2] **Benjamin Roth**[1,3]
[1]*Research Group Data Mining and Machine Learning*,
Faculty of Computer Science, University of Vienna, Vienna, Austria
[2]*UniVie Doctoral School Computer Science*, Vienna, Austria
[3]*Faculty of Philological and Cultural Studies, University of Vienna*, Vienna, Austria
{andreas.stephan,vasiliki.kougia,benjamin.roth}@univie.ac.at

Abstract

In the weakly supervised learning paradigm, *labeling functions* automatically assign heuristic, often noisy, labels to data samples. In this work, we provide a method for learning from weak labels by separating two types of complementary information associated with the labeling functions: information related to the target label and information specific to one labeling function only. Both types of information are reflected to different degrees by all labeled instances. In contrast to previous works that aimed at correcting or removing wrongly labeled instances, we learn a branched deep model that uses all data as-is, but splits the labeling function information in the latent space. Specifically, we propose the end-to-end model *SepLL* which extends a transformer classifier by introducing a latent space for labeling function specific and task-specific information. The learning signal is only given by the labeling functions matches, no pre-processing or label model is required for our method. Notably, the task prediction is made from the latent layer without any direct task signal. Experiments on Wrench text classification tasks show that our model is competitive with the state-of-the-art, and yields a new best average performance.

1 Introduction

In recent years, large language modelling approaches have proven their applicability to a wide range of tasks, mainly due to the pre-training and fine-tuning paradigm. This has created a need for large labeled datasets, as training on these datasets enables models to achieve state-of-the-art performance. However, obtaining manually created labels is expensive, tedious and often requires expert knowledge. As a consequence, significant areas of research are devoted to addressing this challenge by minimizing the need for labeled data. For example, research directions include transfer learning (Ruder et al., 2019) or few-shot learning (Brown et al., 2020). Another research direction to address this challenge is weakly supervised learning. The idea is to use human intuitions, heuristics and existing resources, e.g., related databases, to create weak (noisy) labels.

Several approaches have been proposed to increase the quality of the resulting labels. For example, Ratner et al. (2017) use generative modeling to learn a probability distribution over the labeling function matches, i.e., weak labels, and unknown true labels in order to denoise the labels and subsequently train a classifier. Recently, several works use student-teacher schemes that use knowledge inherent to pre-trained models (Karamanolakis et al., 2021; Cachay et al., 2021; Ren et al., 2020). Usually a summary statistic of weak labels, such as majority vote, is used as ground truth and iteratively updated during training, for example by employing a regularization based on the prediction confidence of the model (Yu et al., 2021). Thus, most methods share the property that the weak labels, i.e., the learning signals, are transformed or updated throughout the learning process.

Instead of updating the weak labels, we want to keep them as-is and make use of a different intuition. Each labeling function provides information relevant to the prediction task but also information only related to the function itself. Our idea is to view these two types of information as complementary and build a model which separates them.

To this end, we propose *SepLL*, an end-to-end model that stacks two branched latent layers, representing target-task-related and labeling-function-related information, on top of a transformer encoder and recombines them for predicting labeling function occurrences (Figure 1). Then, the learning signal is only given by the weak labels. Notably, the task prediction is performed from the latent space without any direct supervision. Multiple information routing strategies are employed to improve the separation.

In order to evaluate the performance, experi-

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3918–3929

ments on the text classification tasks of the Wrench benchmark (Zhang et al., 2021) are performed. Our model achieves state-of-the-art performance when compared to standalone models as well as when combined and compared with the self-improvement method Cosine (Yu et al., 2021). An ablation study shows the importance of each information routing strategy. The experiments show that in addition to its task performance, the model is able to memorize the labeling function information.

The contributions can be summarized in three parts: 1) We introduce a new intuition about the information provided by labeling functions and turn it into a method, *SepLL*, reflecting the intuition in the latent space. 2) We provide an analysis through experiments on the Wrench benchmark, an ablation study and an in depth analysis of the two latent spaces. 3) We provide the code and a suitably transformed version of the input data. [1]

2 Related Work

Weak Supervision. A main concern in machine learning is that a large amount of labeled data is needed in order to train models that achieve state-of-the-art performance. Among others, the field of weak supervision aims to address this issue. The idea is to formalize human knowledge or intuitions into weak supervision sources, called labeling functions, which can be used to produce weak labels. Examples of labeling functions are heuristic rules, e.g., keywords, regular expressions, other pre-trained classifiers or knowledge bases in distant supervision (Craven and Kumlien, 1999; Mintz et al., 2009; Hoffmann et al., 2011; Takamatsu et al., 2012).

A main challenge that appears in a weak supervision setting is how to create accurate labeling functions and how to unify and denoise them. Majority vote, Snorkel (Ratner et al., 2017) (based on data programming) and Flying Squid (Fu et al., 2020) are methods that compute weak labels based on generative models over the labeling function matches and unknown true labels. These models are referred to as label models. Subsequently so called end-models, e.g., BERT-style classifiers (Devlin et al., 2019), or methods dedicated to noisy training labels are used to train a final model.

Recently, neural methods, including the use of pre-trained models, gained more traction. Cachay et al. (2021) use a classifier and a probabilistic encoder for the labeling function matches and optimize them using a noise-aware loss. Similarly, Ren et al. (2020) combine a classifier and a attention-based denoiser, but also include unlabeled samples. Yu et al. (2021) introduced Cosine, which is a method to self-optimize classification models. They leverage contrastive learning and confidence regularization, i.e., high-confidence samples, to optimize a model's performance.

Other approaches use additional signals. For instance, ImplyLoss (Awasthi et al., 2020) uses access to exemplars, i.e., single, correctly labeled samples and ASTRA (Karamanolakis et al., 2021) follows an attention based student-teacher mechanism with an additional supervision of a few manually annotated labeled samples. Zhu et al. (2022) uses a meta self-refinement approach which makes use of access to the validation performance.

Our experiments are built on the Weak Supervision Benchmark (Wrench) (Zhang et al., 2021), which is a framework that aims to provide a unified and standardized way to run and evaluate weak supervision approaches. A wide range of tasks, datasets and implementations of weak supervision methods are available.

Latent Variable Modelling. Existing work regarding latent variable modelling in different areas of machine learning has influenced the rationale behind this work. Research in representation learning has focused on modelling mutually independent factors of variation, e.g., color in computer vision, explicitly in some latent space. Often this is called *disentanglement* (Bengio et al., 2013). This is transferable to our setting as we aim to obtain the task prediction as a disentangled factor. An important early technique is Independent Component Analysis (ICA) (Comon, 1994). Kingma and Welling (2014) introduced variational autoencoders (VAE's) to neural networks, allowing complex data distributions to be represented as simple distributions in the latent space. An extension is given by β-VAE (Higgins et al., 2017), which is more suitable for disentanglement. In addition, there has been progress on theoretical work, which aims to give an insight on what information is identifiable by using self-supervised learning (SSL), e.g., Zimmermann et al. (2021) prove under certain assumptions that it inverts the data generation process. An interesting perspective is the separation of content and style, e.g., the animal in a picture (content) and the camera angle of the image (style). Under milder

[1] https://github.com/AndSt/sepll

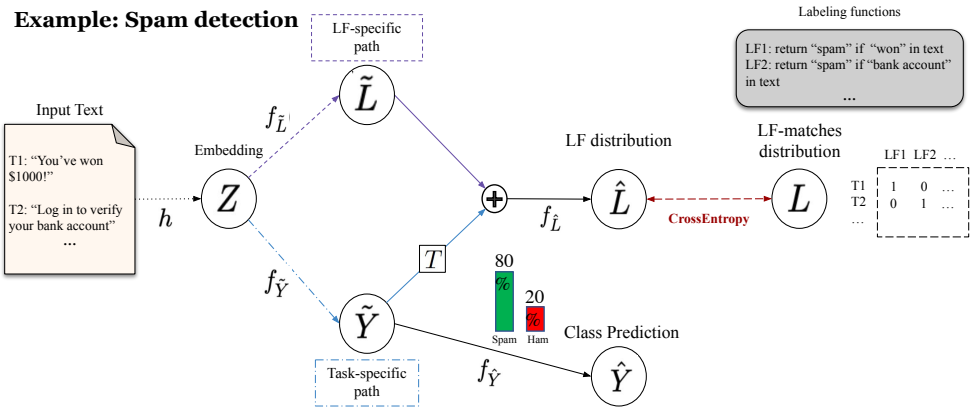


Figure 1: Overview of *SepLL*. Text gets embedded into Z by a Transformer encoder, and then this representation is split into labeling function-specific and task-specific information. The task-specific information is translated back into the LF space and re-combined into $\hat{L}$. A cross-entropy loss between the distribution of labeling function matches L and $\hat{L}$ is minimized. The latent task prediction $\hat{Y}$ can be used for classification.

assumptions as in Zimmermann et al. (2021), it is proved by von Kügelgen et al. (2021) that this separation is achieved using SSL. Mentioned works are not directly applicable to our task, because we want to separate general aspects of labeling functions, which are useful for prediction tasks, from labeling function specific aspects. Another line of research models the true distribution in the latent layer (Sukhbaatar et al., 2015; Goldberger and Ben-Reuven, 2017; Bekker and Goldberger, 2016) while training on the noisy training labels. The typical assumption is that the noise distribution only depends on the class. In weak supervision the noise depends on the input, by definition of labeling functions, thus these types of assumptions are not directly applicable.

3 Method

The motivation of this work is that each labeling function provides two types of information. On the one hand, it provides information about the target task, e.g., spam detection, and on the other hand it provides information related to the labeling function itself. This translates to our model, called *SepLL*, which aims to separate these two types of signals in a latent space. Figure 1 provides an overview of *SepLL*.

In this section, we first introduce some notation and then describe the architecture of *SepLL*. Following, the training mechanisms, which aim to support the separation of the two information types are discussed.

3.1 Problem Setup and Notation

In general, the goal is to solve classification tasks, e.g., spam detection asks whether a text is spam or not. The input space is denoted by X and the unknown labels are denoted by $Y = \{y_1, \ldots, y_c\}$. Additionally m labeling functions $l_i : X \rightarrow \{y\} \cup \emptyset, i = 1, \ldots, m$ are given where each labeling function (LF) either assigns a dedicated specific label $y \in Y$ to a sample or abstains from labeling. If a label is assigned, we say a labeling function *matches* a sample. The task is to use input X and labeling functions l_i to learn a mapping $X \rightarrow Y$. We use the format of the Knodle (Sedova et al., 2021) framework, where each labeling function is encoded as a labeler for exactly one class. This is in contrast to other conventions where a single labeling function is allowed to label multiple classes, e.g., in Ratner et al. (2016). This convention can easily be transformed into our setting, by splitting multi-class LFs into multiple class-specific LFs. The matching matrix $L \in \{0, 1\}^{n \times m}$ describes whether labeling function j matches sample i by setting $L_{ij} = 1$, otherwise $L_{ij} = 0$. The mapping matrix $T \in \{0, 1\}^{m \times |Y|}$ reflects a simple mapping between labeling function i and class j by $T_{ij} = 1$, otherwise $T_{ij} = 0$.

3.2 Basic Model

First, the model transforms input text X into a latent representation Z using an encoder $h : X \rightarrow$

$\mathbb{R}^d$. In this case, a pre-trained transformer encoder transforms input text $x \in X$ into the <CLS>-token embedding $z = h(x) \in \mathbb{R}^d$.

Following, there are two transformations $f_{\tilde{Y}} : \mathbb{R}^d \to \mathbb{R}^{|Y|}$ and $f_{\tilde{L}} : \mathbb{R}^d \to \mathbb{R}^m$, which are realized by two multi-layer perceptrons. The goal is to train the model such that $f_{\tilde{Y}}(x)$ reflects the task information and $f_{\tilde{L}}(x)$ the remaining LF-related information. Note that the resulting output represents the log-space and the transformation to probabilities happens later.

Afterwards, the two latent layers are combined again and compared to the training signal, which is purely given by the LF matches L. The T matrix is used to map the target label information to the corresponding LF information by

$$f_{\hat{L}}(z) = f_{\tilde{Y}}(z)T^\top + f_{\tilde{L}}(z) \in \mathbb{R}^m \qquad (1)$$

where $z = h(x)$ is the latent representation. Crucially, the T matrix establishes the connection between task path and LF path.

In order to run the optimization we compare the combined signals to the labeling functions matches. Therefore we define the *LF distribution* as the normalized L matrix, i.e., $P_{ij} = \frac{L_{ij}}{\sum_k L_{ik}}$.

Finally, the loss is computed as the cross entropy between the labeling function distribution and the prediction

$$\mathrm{CE}(P, Q) = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{m} P_{ij} \log(Q_{ij}) \qquad (2)$$

where the prediction probability Q is given by a softmax activation over $f_{\hat{L}}$.

The task prediction is computed using a softmax activation on the latent task signal, i.e.,

$$P(y_i|x) = \left(f_{\hat{Y}}(z)\right)_i = \frac{e^{\left(f_{\tilde{Y}}(z)\right)_i}}{\sum_{j=1}^{c} e^{\left(f_{\tilde{Y}}(z)\right)_j}} \qquad (3)$$

where $z = h(x)$ is again the latent representation. Thus, no direct supervision is performed.

3.3 Latent information routing

Clearly, the separation could easily collapse by just using the labeling function path, i.e., there is no apparent reason why the LF prediction $\hat{L}$ is not only based on the LF-specific path. Therefore we introduce three schemes supporting the separation of information. Firstly a regularization, secondly an adaption of the learning label and thirdly the inclusion of unlabeled samples are discussed.

Regularization. The easiest solution to the problem is to introduce standard regularization schemes. We consider the L_2 regularization as suitable because it encourages the optimization to put similar weight on the parameters. We test two types of L_2 regularization. Firstly, standard weight decay is used, which employs an L_2 regularization on all parameters, including the transformer encoder. Secondly, an additional L_2 regularization is applied to $f_{\tilde{L}}$. The goal is to regularize the labeling function path such that more weight is put on the class prediction path. In the experimental part we refer to the two types as weight decay and L_2 regularization, respectively.

Noise Injection. Our hypothesis is that the optimization routine puts weight on the task-specific path if two labeling functions belonging to the same class match simultaneously. But most of the time samples are only matched by a single labeling function. Thus, we inject noise in the form of additional *"hallucinated"* matches into the labeling function matrix L. If a sample is matched by a labeling function, we create a match for all other labeling functions for the same class with a probability proportional to a random factor $\lambda \in [0, 1]$, which is a hyperparameter.

Usage of Unlabeled Samples. It has been previously shown, e.g., by Ren et al. (2020) and Yu et al. (2021) that it is effective in weak supervision settings to make use of unlabeled samples X_U, i.e., samples where no labeling function matches. Apart from a performance increase it was shown that the learning gets more robust. Thus, we adapt this semi-supervised learning approach. In order to comply with our basic model, we need to create a labeling function distribution P_U. We take the simplest idea and define P_U as the uniform distribution over the labeling function matches, i.e., $(P_U)_{ij} = \frac{1}{m}$ for all unlabeled samples i and labeling functions j.

4 Experimental Setup

Before analysing the experiments, we discuss the experimental setup. More specifically, an overview of the used datasets, baselines, hyperparameters and some notes on reproducibility and implementational details are given.

Dataset	#Classes	#LF's	# Train	# Coverage	# Dev	# Test
IMDb	2	9	20000	88%	2500	2500
Yelp	2	15	30400	83%	3800	3800
Youtube	2	10	1586	88%	120	250
SMS	2	73	4571	41%	500	500
AGNews	4	9	96000	69%	12000	12000
TREC	6	68	4965	95%	500	500
Spouse	2	9	22254	26%	2811	2701
SemEval	9	164	1749	100%	178	600

Table 1: Statistics describing the datasets as they are used in the WRENCH framework. Coverage is computed on the train set by dividing the number of samples having at least one match by the number of samples.

4.1 Datasets

For the experiments we used eight text classification datasets that are currently included in the Wrench benchmark. As shown in Table 1 the datasets reflect varying properties, such as sample size, coverage or the amount of labeling functions.

Five out of eight datasets represent binary classification problems. These are: (i) **Youtube (Alberto et al., 2015)**, which consists of text comments from YouTube videos, each labeled as spam or non-spam, (ii) **SMS (Almeida et al., 2011)**, which is a mobile phone spam corpus, which contains real SMS messages that are spam or non-spam, (iii) **IMDb review dataset (Maas et al., 2011)** contains reviews from IMDb and each review is labeled as positive or negative, (iv) **Yelp (Zhang et al., 2015)** consists of positive or negative reviews from the Yelp Dataset Challenge 2015, (v) **Spouse (Corney et al., 2016)** , which is a relation classification dataset, where we decide for each sentence if it contains a spouse relation or not. Three datasets correspond to multi-class problems: (i) **AGNews (Zhang et al., 2015)** consists of news articles classified into 4 classes, (ii) **TREC (Li and Roth, 2002)** is a question classification dataset, where the questions are classified into 6 labels, and (iii) **SemEval (Hendrickx et al., 2010)** contains sentences collected from the Web and the task is to identify the relation between two nominals tagged in each sentence among 9 types of semantic relations.

4.2 Baselines

We compare *SepLL* to several traditional and state-of-the-art models that can be categorized in 4 approach types: supervised, statistical, neural and cosine based (see Table 2). Wherever possible the RoBERTa-base (Liu et al., 2020) backbone model is used.

For the supervised approach (**Gold + RoBERTa**), which serves as an upper bound, we perform a standard fine-tuning of a RoBERTa (Liu et al., 2020) pre-trained model using gold labels.

The statistical approaches are: (i) **Majority Vote (MV)** As the name suggests, majority vote picks the label indicated by the majority of labeling function matches. In case there is no match, a random label is chosen, (ii) **Data Programming (DP)** employs Snorkel DP (Ratner et al., 2017) to obtain weak labels and (iii) **Flying Squid (FS)** (Fu et al., 2020) is a label model making use of so-called triplet methods.

In the neural category of baselines there are three dedicated end-to-end models: (i) **WeaSEL** (Cachay et al., 2021) uses a classifier and a probabilistic encoder combined with a noise-aware loss function, (ii) **Denoise** (Ren et al., 2020) uses a classifier and an attention-based denoiser, mutually optimizing each other, (iii) **KnowMAN** is an adversarial architecture that aims to learn representations that are invariant to the labeling function signals (März et al., 2021). In addition to learning a classifier for the end task, a labeling function discriminator is trained and its negative gradient is used to update a shared feature extractor.

We also employ models that combine the labels obtained by MV, DP and FS with standard supervised learning using the **RoBERTa** model, which we call **RoBERTa-MV**, **RoBERTa-DP** and **RoBERTa-FS** respectively (Liu et al., 2020).

Cosine (Yu et al., 2021) takes a pre-trained classifier and uses contrastive learning and confidence regularization to improve the performance of the classifier. Cosine is a model-agnostic method for self-training that can be combined

	IMDb	Yelp	Youtube	SMS	AGNews	Trec	Spouse	Semeval	Avg.
Supervised									
Gold+RoBERTa†	93.25	97.13	95.68	96.31	91.39	96.68	-	93.23	88.58
Statistical									
MV†	71.04	70.21	84.00	23.97	63.84	60.80	20.81	77.33	59.00
DP† (Ratner et al., 2017)	70.96	69.37	82.00	23.78	63.90	64.20	21.12	71.00	58.29
FS† (Fu et al., 2020)	70.36	68.68	76.80	0.00	60.98	31.40	34.3	31.83	46.79
Neural									
WeaSEL (Cachay et al., 2021)	85.16	91.23	96.40	2.94	85.92	64.2	0.00	44.30	58.77
Denoise† (Ren et al., 2020)	76.22	71.56	76.56	91.69	83.45	56.20	22.47	80.83	69.87
KnowMAN (März et al., 2021)	59.00	76.76	94.00	92.80	84.68	65.20	25.48	80.50	72.30
RoBERTaMV†	85.76	89.91	96.56	94.17	**86.88**	66.28	17.99	84.00	77.69
RoBERTaDP†	86.26	89.59	95.60	28.25	86.81	72.12	17.62	70.57	68.35
RoBERTaFS†	**86.95**	**92.08**	93.84	10.72	86.69	30.44	0.0	31.83	54.07
SepLL	83.57	91.32	**97.50**	**95.45**	85.47	**81.25**	**43.2**	**87.33**	**83.14**
+Cosine (Yu et al., 2021)									
RobertaMV+Cosine†	**88.22**	94.23	**97.60**	96.67	**88.15**	77.96	40.5	86.20	83.69
RobertaDP+Cosine†	87.91	94.09	96.80	31.71	87.53	82.36	28.86	75.17	73.05
SepLL+Cosine	88.00	**95.07**	**97.60**	**97.01**	86.28	**83.40**	**43.37**	**86.83**	**84.70**

Table 2: Results on the Wrench benchmark tasks. Accuracy values are reported for all datasets, except for SMS and Spouse where the binary F_1 is shown due to class imbalance. Numbers directly taken from the Wrench paper are marked by †.

with any classifier and is not specific to weak supervision. It has been observed that Cosine particularly helps with standard weak supervision methods like majority voting and data programming, and we include the best-performing combinations from Wrench (**RoBERTa-MV+Cosine**, **RoBERTa-DP+Cosine**), as well as *SepLL*+Cosine in our experiments. Additional information on the baselines is given in appendix A.

4.3 Hyperparameters

Two types of hyperparameters are tuned, namely transformer related hyperparameters and information routing related ones. We perform a grid search on these and take the best model based on the validation set. The first group includes a learning rate in {1e-5, 2e-5, 5e-5}, a batch size of 16 and warmup steps in $\{0, 100\}$. For information routing related hyperparameters we search for weight decay in $\{0, 0.01, 0.001\}$, L_2-regularization on the LF path in $\{0.1, 0.5, 0., 1.\}$, and for noise injection $\lambda \in \{0., 0.1, 0.2, 0.05\}$. *SepLL* is always trained using AdamW (Loshchilov and Hutter, 2019).

4.4 Implementation and Reproducibility

Our implementation is in JAX (Bradbury et al., 2018), using the Huggingface transformers library (Wolf et al., 2020) as a high level interface.

In order to save resources, we report numbers directly from the Wrench paper, whenever applicable. In other cases, we use the datasets and the evaluation setting of Wrench, and the original implementations and reported hyper-parameters of the respective publications.

We use RoBERTa-base as backbone of most models, so all models use approximately 125 million parameters. Fine-tuning one instance takes between 5 and 60 minutes, depending on the dataset size. The experiments are performed on a Nvidia DGX-1 using a single Tesla V100 graphics card per run.

5 Experiments

This section provides an analysis of the capabilities of the model. After the general performance analysis, an ablation study of the information strategies is performed. Furthermore, we investigate how much information flows through the path associated with the latent class prediction, and how much though the other, LF-specific path.

5.1 General Performance

The results are split into two parts. Firstly, a comparison with the standard baselines is given. Secondly, we analyse the impact of Cosine separately, as we view it as a general framework to self-optimize the prediction performance of any pre-trained classifier. The results are shown in Table 2. *SepLL* outperforms the standard baselines on all tasks except IMDb and AGNews. We think

	avg
Full Model	83.14
– Weight decay	82.73
– L_2-reg.	82.59
– Unlabeled data	82.11
– Noise injection	81.77
Basic model	80.85

Table 3: Ablation of information routing strategies. Each strategy is removed individually. The basic model does not use any strategy. The average is taken over all datasets on the test set.

this is due to the fact that both datasets are rather large while having a low number of labeling functions. The same trend is observable when combined with Cosine. On most tasks a new state-of-the-art is reached. A negative exception is given by SemEval, where Cosine is not able to generalize from our base model. Importantly, the average performance over all tasks is improved by a margin of 6% on the standard baselines and 1% in the Cosine section, relative to the respective best-performing comparison method.

5.2 Ablation of Information Routing Strategies

In order to understand the impact of the routing strategies we perform an ablation study, which is shown in Table 3. Starting from the full model, each routing strategy is removed individually, and in the last row all strategies are removed. The average performance over all datasets is shown. Given that there are different types of metrics this is just an aggregate view of the performance – a detailed ablation including the performance per task is given in Appendix B.

We observe that each strategy helps the performance of the end model and that the noise injection strategy has the highest positive impact. This reinforces the initial assumption that a sample has a larger learning impact on the task-specific path if multiple labeling functions belonging to one class match simultaneously.

Nevertheless, the basic model (without active routing) performs decent in comparison to the baselines in Table 2. This is surprising because there is no apparent incentive to prevent the model to collapse to the LF path. One possible explanation is that the gradient updates nevertheless flow through both paths and still update the task path in a way that it is consistent with the LF-prediction. Moreover, predicting the LF through the task-path, albeit less accurate, is more parameter efficient than going through the LF-path.

5.3 Labeling Function Memorization

As a by-product of the architecture, it is possible to predict whether a labeling function matches or not. As these matches represent the learning signal, an evaluation of the prediction of labeling function matches also shows how much information is retained. In Figure 2 multiple metrics are computed to analyze the memorization of matches. The accuracy and the F_1-score (because the matrix L is sparse) for LF-predictions, averaged over all test sets, are computed to measure memorization of the labeling functions matches.

In order to transform logits into predictions, we compute the softmax activation on the functions $f_{\tilde{L}}$ and $f_{\hat{L}}$, and define the prediction as

$$L_{ij} = 1 \Leftrightarrow \text{softmax}\left(f_*(z_i)\right)_j > \frac{4}{m}$$

otherwise $L_{ij} = 0$, i.e., if a sample has 4 times the probability of the uniform distribution. We tested the threshold $k = 2, 3, 4$ on the dev set of IMDb and TREC and took the best performing one. Note that $k = 3, 4$ performed almost equal. To compute the task predictions, we apply the same threshold to the softmax of the mapped task logits, i.e., softmax $\left(f_{\tilde{Y}}(z_i)T^\top\right)$. We call the outputs of the softmax prediction distribution. The diagram on the right in Figure 2 displays the cross-entropy between the LF distribution P (see section 3) and the prediction distributions, and the uniform distribution $U[0, 1]$.

The results show that the prediction from the final output and from the LF path are nearly identical. Interestingly, the latent LF path achieves a slightly better accuracy, but slightly worse F_1-score. In comparison to the full and the latent model, the cross-entropy between $\tilde{Y}$ and P is high where the latter approaches the cross-entropy between $\hat{L}$ and the uniform distribution. These results indicate that $\tilde{Y}$ does not substantially influence the prediction of L. Still, the prediction made from the task-related path $f_{\tilde{Y}}$ reaches an average F_1-score of 88.5%, which shows that it still contains much of the information needed to predict LF matches. In appendix C an additional figure shows that the difference between performance on the training set and the

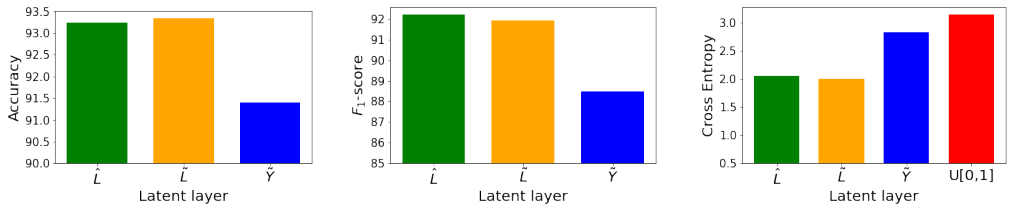


Figure 2: The figure shows the accuracy (left), the F_1-score (center) and the cross-entropy (right) between true labels L and the predictions based on the full model $f_{\hat{L}}$ and the latent representations $f_{\tilde{Y}}, f_{\tilde{L}}$ predictions. On the right an additional comparison to the uniform distribution is presented.

test set is low, indicating that there is little to no overfitting towards the training LF distribution.

5.4 Impact of Number of LF Matches

One could expect that it is easier to predict the class label for samples with many labeling functions matches. Figure 3 shows the performance of each task on the test set in relation to the number of labeling function matches (note that we use those LF-matches only for analysis purposes, they are not observed by the model) . The figure does not contain SemEval as it has exactly one match per sample.

Interestingly, the performance on samples with no LF-match is, for most datasets (apart from Spouse), almost on par with samples that include patterns from one or more LFs. It is therefore fair to say that the model generalizes beyond the labeling function information.

An exception is given by the Spouse dataset, which has the lowest coverage in the training set (see Table 1). Appendix D provides additional numbers, including a table which shows the amount of samples per number of LF-matches per dataset.

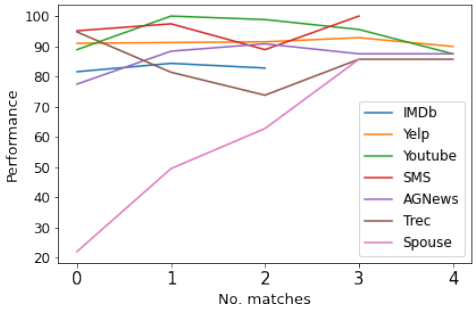


Figure 3: Performance (accuracy or F1 score) on the tasks given the amount of labeling function matches (test set).

6 Conclusion

This work tackles weak supervision in a novel way. Instead of denoising the weak labels or iteratively updating them, the weak labels are used as the training target as-is. We introduce the *SepLL* model, which separates information relevant for the target task through the usage of two latent spaces. One latent space is used to perform downstream task prediction, the other one aims to keep the labeling function specific information. Thus, the prediction is made from the latent space without any direct supervision. Experiments show that the model is able to achieve state-of-the-art performance. An additional investigation shows that the model is able to memorize the labeling function information. Hence, the model cannot only be used for downstream tasks but also to predict labeling function matches.

7 Limitations

As in the experiments in the Wrench benchmark, we use validation data for early stopping. If a setting was purely weakly supervised, with no annotated data, this would not be possible. To ensure comparability, we stick to the Wrench setup, which assumes that the cost of annotating a small sample size for validation is acceptable in most scenarios.

Weak supervision performance highly depends on the quality and other properties of the labeling functions. In contrast to more standardized scenarios of machine learning, e.g. in supervised learning, where it can be assumed that training and test data samples are drawn i.i.d. from the same distribution, such arguments cannot be made about the labeling functions and their relationship to the correct annotations.

Formal analysis, as it has been attempted some-

times for weak supervision, relies on heavy assumptions, which are typically unrealistic and likely to be wrong. Such assumptions could be: the weak labelers produce noise following a defined noise distribution; the weak annotations are independent given the class label; each weak labeler exceeds a threshold accuracy; etc. In this paper, we do not attempt a formal analysis based on such assumptions. However, we compare to other methods which are based on such assumptions, and we outperform them in our experiments. With other data sets, potentially showing other characteristics (other tasks, other languages, other labeling functions), the performance could be different. In particular, since the weakly supervised method performs almost as well as the supervised model, there is a need for tasks and datasets that are more challenging.

Another limitation is that currently our model is only implemented for classification, not for sequence tagging. This adaptation would be possible, but is not trivial since sequential dependencies, unlabeled tokens, search, etc. would need to be carefully handled. We leave these extensions for future work.

Acknowledgements

This research was funded by the WWTF through the project "Knowledge-infused Deep Learning for Natural Language Processing" (WWTF Vienna Research Group VRG19-008).

References

T C Alberto, J V Lochter, and T A Almeida. 2015. TubeSpam: Comment Spam Filtering on YouTube. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 138–143.

Tiago A Almeida, J M G Hidalgo, and A Yamakami. 2011. Contributions to the study of SMS spam filtering: new collection and results. In *DocEng '11*.

Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. 2020. Learning from Rules Generalizing Labeled Exemplars.

Alan Joseph Bekker and Jacob Goldberger. 2016. Training deep neural-networks based on unreliable labels. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2682–2686, Shanghai, China.

Y Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35:1798–1828.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. JAX: composable transformations of Python+NumPy programs.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners.

Salva Rühling Cachay, Benedikt Boecking, and Artur Dubrawski. 2021. End-to-End Weak Supervision. In *Advances in Neural Information Processing Systems*.

Pierre Comon. 1994. Independent component analysis, A new concept? *Signal Processing*, 36(3):287–314.

D Corney, M-Dyaa Albakour, Miguel Martinez-Alvarez, and Samir Moussa. 2016. What do a Million News Articles Look like? In *NewsIR@ECIR*.

Mark Craven and Johan Kumlien. 1999. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology, August 6-10, 1999, Heidelberg, Germany*, pages 77–86. AAAI.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Daniel Y. Fu, Mayee F. Chen, Frederic Sala, Sarah M. Hooper, Kayvon Fatahalian, and Christopher Ré. 2020. Fast and three-rious: Speeding up weak supervision with triplet methods. In *ICML*, pages 3280–3291.

Jacob Goldberger and Ehud Ben-Reuven. 2017. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations*, Toulon, France.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38, Uppsala, Sweden. Association for Computational Linguistics.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017.

beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *ICLR*.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 541–550.

Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan Awadallah. 2021. Self-Training with Weak Supervision.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Ro{BERT}a: A Robustly Optimized {BERT} Pretraining Approach.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Luisa März, Ehsaneddin Asgari, Fabienne Braune, Franziska Zimmermann, and Benjamin Roth. 2021. KnowMAN: Weakly supervised multinomial adversarial networks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9549–9557, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics.

Alexander Ratner, Stephen H Bach, Henry R Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid Training Data Creation with Weak Supervision. *CoRR*, abs/1711.1.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. 2016. Data programming: Creating large training sets, quickly. *Advances in neural information processing systems*, 29.

Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. 2020. Denoising Multi-Source Weak Supervision for Neural Text Classification. *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. 2019. Transfer Learning in Natural Language Processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, Minneapolis, Minnesota. Association for Computational Linguistics.

Anastasiia Sedova, Andreas Stephan, Marina Speranskaya, and Benjamin Roth. 2021. Knodle: Modular Weakly Supervised Learning with PyTorch. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 100–111, Online. Association for Computational Linguistics.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. 2015. Training convolutional networks with noisy labels. 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. 2012. Reducing Wrong Labels in Distant Supervision for Relation Extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 721–729, Jeju Island, Korea. Association for Computational Linguistics.

Julius von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. 2021. Self-Supervised Learning with Data Augmentations Provably Isolates Content from Style.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M Rush. 2020. HuggingFace's Transformers: State-of-the-art Natural Language Processing.

Yue Yu, Simiao Zuo, Haoming Jiang, Wendi Ren, Tuo Zhao, and Chao Zhang. 2021. Fine-Tuning Pre-trained Language Model with Weak Supervision: A Contrastive-Regularized Self-Training Approach.

Jieyu Zhang, Yue Yu, Yinghao Li, Yujing Wang, Yaming Yang, Mao Yang, and Alexander Ratner. 2021.

WRENCH: A Comprehensive Benchmark for Weak Supervision.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Dawei Zhu, Xiaoyu Shen, Michael A Hedderich, and Dietrich Klakow. 2022. Meta Self-Refinement for Robust Learning with Weak Supervision. *arXiv preprint arXiv:2205.07290*.

Roland S Zimmermann, Yash Sharma, Steffen Schneider, Matthias Bethge, and Wieland Brendel. 2021. Contrastive Learning Inverts the Data Generating Process. In *ICML*.

A Experimental Description

As mentioned in the main paper, the code, setup and hyperparameters for the baselines are taken from the Wrench benchmark and are described in the corresponding paper (Zhang et al., 2021). For the Gold+RoBERTa model there was no result available, as there are no ground truth labels for train set. Thus we assume an F_1-score of 45 for the computation of the average as it is better than the best performing model.

In the following paragraphs, the hyperparameters for the baselines we trained ourselves are briefly presented.

Weasel (Cachay et al., 2021). We experiment with hyperparameters similar to the original paper because there no transformer encoder "roberta-base" is used. We use AdamW (Loshchilov and Hutter, 2019) and optimize hyperparameters for temperature in $\{0.33, 1.0\}$, dropout in $\{0.1, 0.2\}$, weight decay in $\{1^{-4}, 7^{-7}\}$ and learning rate in $\{1^{-4}, 1^{-5}, 2^{-5}\}$. Unfortunately we were unable to obtain reasonable performance on the skewed datasets SMS and Spouse. The original paper does not use pre-trained language models, thus no direct comparison is possible. We decided to take the same encoder for all models.

KnowMAN (März et al., 2021). The values are set similar to the original paper. Batch size is 16, hidden size 700, dropout is 0.4 and trained is up to 5 epochs. The classifier(C) and the discriminator(D) use Adam (Kingma and Ba, 2014) with a learning rate of $1^{-}4$ and the feature extractor(F) uses AdamW (Loshchilov and Hutter, 2019) with the same learning rate. For all three, *num_layer* is set to 1.

B Ablations

In addition to the summarized ablation in section 5.2, Table 4 shows the impact of the ablations on task level. We observe that in general, the results for each task agree with the average.

C Labeling Function Memorization

The detailed numbers corresponding to the evaluation of section 5.3 are given in Table 5. The absolute difference between train and test split is shown in Figure 4. The differences are low for accuracy, F_1-score and cross-entropy, thus we conclude that there is no or rather little overfitting towards the train set.

D Impact of number of matches

Table 5 provides detailed numbers for Figure 3. Since Figure 3 does not reflect how many samples correspond to a certain number of LF matches, detailed counts are added in Table 7. This is an important addition as a performance measurement on a small number of samples is not indicative of the true performance. Luckiliy, usually the number of matching samples per number of matching labeling functions is distributed rather well.

	IMDb	Yelp	Youtube	SMS	AGNews	Trec	Spouse	SemEval	Avg.
Full model	83.57	91.32	97.50	95.45	85.47	81.25	43.20	87.33	83.14
– Weight decay	83.57	91.32	97.50	95.45	85.47	79.03	42.55	86.99	82.73
– L_2 reg.	83.25	91.14	97.08	94.57	85.47	79.03	43.20	86.99	82.59
– Unlabeled data	82.13	88.34	97.50	95.45	85.32	81.25	39.89	86.99	82.11
– Noise injection	82.13	88.34	97.50	95.45	85.46	78.43	39.89	86.99	81.77
Basic model	82.13	86.89	97.08	93.85	85.14	74.80	39.89	86.99	80.85

Table 4: Extension of the ablation in Table 3, showing all datasets.

Dataset	$\tilde{L}$ Acc.	$\hat{L}$ Acc.	$\tilde{Y}$ Acc.	$\tilde{L}$ F_1	$\hat{L}$ F_1	$\tilde{Y}$ F_1	$\tilde{L}$ CE	$\hat{L}$ CE	$\tilde{Y}$ CE	$U[0,1]$ CE
IMDb	90.21	90.24	86.88	89.41	89.58	80.77	1.42	1.41	2.11	2.20
Yelp	91.44	91.46	89.94	89.46	89.51	85.18	2.19	2.13	2.58	2.71
Youtube	82.80	83.00	81.60	76.32	77.03	73.33	1.72	1.70	2.16	2.30
SMS	96.41	95.65	99.28	97.59	97.19	98.92	4.26	5.31	4.76	4.29
AGNews	91.05	91.54	89.48	89.06	90.45	84.50	1.72	1.66	1.92	2.20
Trec	99.42	99.23	95.49	99.42	99.24	95.90	1.11	1.05	3.59	4.22
Spouse	95.98	96.32	95.81	94.64	95.89	93.76	2.03	2.01	2.14	2.20
SemEval	99.47	98.45	92.70	99.52	98.81	95.55	1.55	1.21	3.36	5.10
Avg.	93.35	93.24	91.40	91.93	92.21	88.49	2.00	2.06	2.83	3.15

Table 5: Extension of Figure 2, showing all numbers explicitly. The presented metrics are accuracy, macro F_1-score and cross entropy (CE) between true LF matrix L (see section 3) and the rows which are the full model $f_{\hat{L}}$, the latent layers $f_{\tilde{L}}, f_{\tilde{Y}}$ and the uniform distribution $\mathcal{U}[0,1]$.

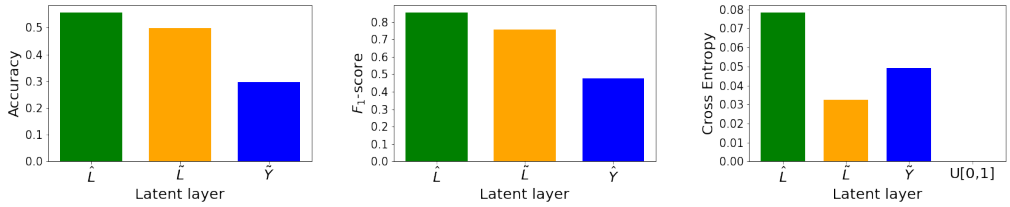


Figure 4: This is extension to Figure 2, describing the same metrics. But here we compute the the absolute difference between the metrics computed on the train set and the test set.

	0	1	2	3	4	5
IMDb	81.58	84.35	82.80	0.00	0.00	0.0
Yelp	91.00	91.25	91.48	92.84	89.93	62.5
Youtube	88.89	100.00	98.84	95.56	87.50	0.0
SMS	95.12	97.44	88.89	100.00	0.00	0.0
AGNews	77.48	88.38	90.81	87.50	87.50	0.0
Trec	94.74	81.40	73.81	85.71	85.71	0.0
Spouse	22.06	49.51	62.77	85.71	0.00	0.0
SemEval	0.00	85.09	0.00	0.00	0.00	0.0

Table 6: Detailed numbers corresponding to Figure 3. The columns describe the number of matches a sample has, the cells the performance, i.e. accuracy or F_1-score.

#matches	total	0	1	2	3	4	5
IMDb	2953	304	1521	593	0	0	0
Yelp	5732	611	1463	1092	475	139	16
Youtube	460	18	86	86	45	8	0
SMS	263	302	148	37	12	0	0
AGNews	11367	3699	5622	2318	336	24	0
Trec	788	19	301	84	70	21	0
Spouse	1019	1951	519	196	33	1	0
SemEval	764.0	0	436	0.0	0.0	0.0	0.0

Table 7: It is shown how many samples there are in total and split up in number of matches per sample. Numbers are computed on the test set.

7. Text-Guided Image Clustering

Title	Text-Guided Image Clustering
Authors	Andreas Stephan, Lukas Miklautz, Kevin Sidak, Jan Philip Wahle, Bela Gipp, Claudia Plant and Benjamin Roth
Publication Outlet	In Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2960–2976, St. Julian's, Malta. Association for Computational Linguistics.
Copyright Information	
Thesis-Reference	Stephan et al. (2024b)

Supplementary Material

The code is available on Github at https://github.com/AndSt/text_guided_cl.

Author Contributions

Andreas Stephan	conceived the initial idea, developed and expanded on the initial idea. He implemented the code, designed the experiments, conducted the experiments, analyzed them, and iterated on them. Further, he wrote the majority of the paper.
Lukas Miklautz	participated in developing and expanding the initial idea, provided feedback on the modeling, experimental design, and experiments, and helped write the paper.
Kevin Sidak	participated in developing and expanding the initial idea, provided feedback on the modeling and experimental design.
Jan Philip Wahle	provided feedback on the modeling, the experimental design, and experiments, and helped write the paper.
Bela Gipp	provided feedback on the modeling, the experimental design, and experiments, and in general, provided mentoring and supervision.
Claudia Plant	provided feedback on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.
Benjamin Roth	participated in developing and expanding the initial idea, provided feedback on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.

Text-Guided Image Clustering

Andreas Stephan[1,2,5], **Lukas Miklautz**[1,2], **Kevin Sidak**[1,2], **Jan Philip Wahle**[4], **Bela Gipp**[4], **Claudia Plant**[1] and **Benjamin Roth**[1,3]
[1] Faculty of Computer Science, University of Vienna, Austria
[2] UniVie Doctoral School Computer Science, University of Vienna, Austria
[3] Faculty of Philological and Cultural Studies, University of Vienna, Austria
[4] Georg-August-Universität Göttingen, Germany
[5] andreas.stephan@univie.ac.at

Abstract

Image clustering divides a collection of images into meaningful groups, typically interpreted post-hoc via human-given annotations. Those are usually in the form of text, begging the question of using text as an abstraction for image clustering. Current image clustering methods, however, neglect the use of generated textual descriptions. We, therefore, propose *Text-Guided Image Clustering*, i.e., generating text using image captioning and visual question-answering (VQA) models and subsequently clustering the generated text. Further, we introduce a novel approach to inject task- or domain knowledge for clustering by prompting VQA models. Across eight diverse image clustering datasets, our results show that the obtained text representations often outperform image features. Additionally, we propose a counting-based cluster explainability method. Our evaluations show that the derived keyword-based explanations describe clusters better than the respective cluster accuracy suggests. Overall, this research challenges traditional approaches and paves the way for a paradigm shift in image clustering, using generated text[1].

1 Introduction

Psychologists, neuroscientists, and linguists have long studied the dependence of vision and language in humans (Pinker and Bloom, 1990; Nowak et al., 2002; Corballis, 2017). Although the relationship between these modalities is not fully understood, there is a consistent finding: the brain generates a condensed representation to transmit visual information between brain regions (Cavanagh, 2021). A widely discussed type of representation is often referred to as "visual language" or "language of thought" (Fodor, 1975; Jackendoff et al., 1996). Studies based on these concepts suggest that language can be a crucial driver of visual understanding. For example, children remember conjunctions of visual features better when accompanied by a textual description (Dessalegn and Landau, 2013), e.g., "the yellow is left of the black". Given this relationship between visual perception and language comprehension, the question arises whether an abstract textual representation benefits image clustering.

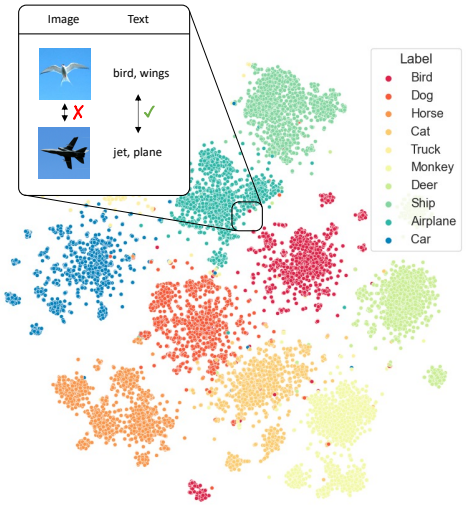


Figure 1: A t-SNE visualization of the BLIP-2 image embeddings for the STL10 dataset. While the images are highly similar (blue background), text such as bird and jet clearly distinguishes objects (and clusters).

With the significant growth of visual content created online, image clustering has become essential in, e.g., retrieval systems, image segmentation, or medical applications (Mittal et al., 2021; Pandey and Khanna, 2016; Kart et al., 2021). Language offers dense, human-interpretable information, providing multiple benefits when clustering (Figure 1). Emerging multi-modal foundation models and large language models (LLMs), e.g., Blip2 (Li et al., 2023) or GPT-3 (Davidson et al., 2018), allow to derive a "visual language" from images.

[1] Github repo: https://github.com/AndSt/text_guided_cl

Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics
Volume 1: Long Papers, pages 2960–2976

In this paper, we propose *text-guided image clustering*, i.e., deriving a textual representation from images to perform clustering purely based on their text representation. In Figure 2, we outline three approaches to text-guided image clustering. These approaches are structured by the degree of external knowledge introduced into the clustering process.

First, *caption-guided clustering* uses image captioning models to generate brief descriptions of the image content, requiring no external knowledge. In order to inspect the qualities of image and text representations, we compare vision encoder embeddings with TF-IDF (Sparck Jones, 1972) and SentenceBERT (SBERT, Reimers and Gurevych, 2019) representations of the generated text. Our experiments show that on a broad set of eight image clustering datasets, text representations on average outperform the image representations of three state-of-the-art (SOTA) models. Second, *keyword-guided clustering* injects knowledge about the clustering task by prompting visual question-answering (VQA) models to generate keywords, using the assumption that only a few keywords of interest are necessary to describe each image sufficiently. Interestingly, we observe an average performance increase of 5% for TF-IDF-based clusterings. Third, *prompt-guided clustering* introduces domain knowledge in the form of tailored prompts for VQA models. Quantitatively, we observe another performance increase and qualitatively show that clusters related to the question are formed better. Further, we propose to use the generated text for a straightforward counting-based cluster explainability method, generating a keyword-based description for each cluster.

Our contributions can be summarized as follows:

- We propose text-guided image clustering, a novel paradigm leveraging generated text for image clustering.
- We introduce a new way of image clustering by injecting task- and domain knowledge via prompting visual question-answering models.
- We show in our experiments that text-guided image clustering is competitive and often outperforms clustering solely based on images on several datasets.
- We propose a counting-based method to generate a description for each cluster, often exhibiting stronger interpretability than the cluster accuracy suggests.

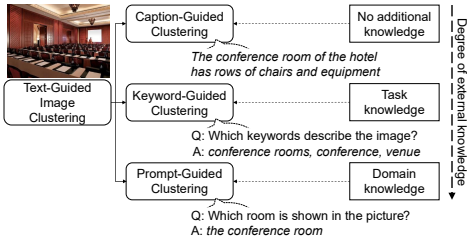


Figure 2: Taxonomy of the text generation processes, structured by the degree of external knowledge. Text is generated BLIP-2 (Li et al., 2023).

2 Related Work

We approach image clustering in a novel way by generating more abstract text descriptions using image-to-text models. Therefore, we discuss how our approach relates to earlier work in image clustering (Section 2.1), text clustering (Section 2.2) and give an overview of the enabling technology of image-to-text models in Section 2.3.

2.1 Image Clustering

Clustering is the task of grouping similar objects together while keeping dissimilar ones apart. A key problem for unsupervised clustering of images is finding a good similarity measure. Deep learning-based clustering methods approach this problem by learning a representation that maps semantically similar images closer together (Xie et al., 2016; Yang et al., 2017; Niu et al., 2020; Caron et al., 2018; Zhou et al., 2022b). A downside of unsupervised methods is that relying only on image information can suffer from the *blue sky problem* (Häusser et al., 2018). For example, in Figure 1, the blue background pixels make up most of the images. Our approach circumvents this downside by generating a concise textual description of an image. Multi-view clustering methods like (Jin et al., 2015; Chaudhary et al., 2019; Yang et al., 2021; Xu et al., 2022) combine heterogeneous views of data instances into a single clustering. In contrast to our work, they assume the availability of all modalities.

An important problem in clustering is explainability (Fraiman et al., 2011; Moshkovitz et al., 2020), aiming to describe the content of the individual clusters. In general, there are clustering algorithms designed such that the resulting clustering is explainable (Dao et al., 2018), or post-processing methods that explain a given clustering. Existing methods use interpretable features such as

semantic tags (Sambaturu et al., 2020; Davidson et al., 2018), especially when textual explainability is considered. For instance, Zhang and Davidson (2021) uses integer linear programming to assign tags to clusters. Contrary to our approach, these methods assume given textual tags.

2.2 Text Clustering

Typically, the text is transformed into a vector representation, and then a clustering algorithm, e.g., K-Means, is applied. Early text representation approaches use counting-based representations such as Bag-of-Words (BoW) or TF-IDF (Sparck Jones, 1972; Zhang et al., 2011). The field moved away from frequency-based approaches as they neglect word order and cannot represent contextualized information, e.g., computer 'mouse' vs. the animal 'mouse' (Peters et al., 2018). In recent years, the focus in Natural Language Processing (NLP) shifted towards contextualized neural network-based vector encodings, dominated by transformer-based methods (Vaswani et al., 2017). The first breakthrough in transformer-based sentence representations was Sentence-BERT (SBERT) (Reimers and Gurevych, 2019), a siamese network architecture fine-tuning BERT (Devlin et al., 2019) on supervised datasets, e.g. NLI. Following SBERT, text representation techniques are mostly trained using contrastive learning where the choice of positive and negative pairs is unsupervised, e.g., SimCSE (Gao et al., 2021), or weakly-supervised, e.g., E5 (Wang et al., 2022b).

2.3 Image-To-Text Models

Image captioning provides textual descriptions for given images. NIC (Vinyals et al., 2015) introduces the now common use of an image encoder and a language decoder. Subsequent models (Radford et al., 2021; Yuan et al., 2021) additionally allow multi-modal inputs, integrating both image and textual information to improve captioning and support tasks like Visual Question Answering (VQA) (Antol et al., 2015). Wang et al. (2022a) use only one image encoder and one text decoder, and perform image /video captioning and VQA in one simplified architecture. Flamingo (Alayrac et al., 2022) allows interleaving images and text by introducing Perceiver Resamplers on top of pre-trained image and language models. BLIP-2 (Li et al., 2023) is a state-of-the-art model that takes fixed pre-trained language and image models and only fine-tunes a so-called Query-Transformer, which only uses a few trainable parameters. This is useful in our experiments because the underlying models are not trained on multimodal data, ensuring a fair comparison of the respective representations.

3 Methodology

We describe the formal setup, the experimental setup, and the chosen datasets.

3.1 Problem Definition

Let $\mathbf{X} = \mathbf{x}_1, \cdots, \mathbf{x}_n \subset \mathcal{X}$ denote the set of images in our dataset. The goal of image clustering is to obtain a clustering $h : \mathcal{X} \to \mathcal{Y}$ that assigns images to their respective clusters. We propose to employ image-to-text models which typically consist of an image encoder $f : \mathcal{X} \to \mathcal{Z}$, embedding images into a latent space $\mathcal{Z} \subset \mathbb{R}^d$, and a text decoder, i.e. a LLM, $g : \mathcal{Z} \to \mathcal{T}$, where $\mathcal{T}$ is some text space. The text is subsequently embedded $t : \mathcal{T} \to \mathcal{V} \subset \mathbb{R}^l$ and clustered, e.g., with K-Means.

3.2 Experimental Setup

The central goal of this paper is to compare representations based on images and generated text for the task of image clustering. The following describes the choices and evaluation criteria common to all experiments.

Clustering. To shed light on the question of whether text is a (more) suitable representation for image clustering, we compare the performance of a clustering on the image space $\mathbf{Z} = f(\mathbf{X})$ and of a clustering on the vectorization of the generated text $\mathbf{T} = t(g(\mathbf{Z}))$. Following the deep clustering (Xie et al., 2016; Yang et al., 2017) and self-supervised learning (Zhou et al., 2022a) literature, we use K-Means to evaluate the suitability of the respective image and text embeddings for clustering. We run K-Means 50 times in all experiments and report the mean outcome to get robust results. Whenever we need a single run, e.g., for qualitative analysis, the run with the lowest K-Means loss, also called inertia, is used.

Vectorization. In order to employ clustering algorithms, images and texts need to be represented as vectors. For image vectorization, we use the latent space of an image encoder. We experiment with multiple models introduced in Section 4.1. For text vectorization, one frequency-based and one neural algorithm are considered. TF-IDF (Sparck Jones, 1972) is a standard counting-based representation. Using the scikit-learn (Pedregosa et al., 2011) im-

plementation, English stop-words are removed, and a maximum vocabulary of 2000 words is set. No additional preprocessing is performed. Since nowadays transformer-based text representations are the standard, we experiment with SBERT[2] (Reimers and Gurevych, 2019) as it was the first BERT-based sentence representation. Note that larger, newer, and better transformer-based models are available. We deliberately choose a widely used, competitive, small model as this strengthens our claim that clusterings based on generated text often outperform clusterings based on image representations.

Metrics. To measure clustering performance, the Normalized Mutual Information (NMI) (Vinh et al., 2010) and the Cluster Accuracy (Acc) (Yang et al., 2010) are computed. Both metrics take values between 0 and 1, where higher numbers indicate a better match with the ground truth labels. For the sake of readability, we multiply them by 100.

3.3 Datasets

We consider a diverse collection of datasets, separated into three groups according to various challenges. Partially, there is an overlap between the properties of the datasets. Nevertheless, our selection of datasets is motivated by this grouping. Note that this is a more diverse set of datasets as typically used (Cai et al., 2022; Qian, 2023). An overview of the dataset statistics and samples of each dataset are depicted in Tables 6 and 7 in the Appendix, respectively.

Standard Datasets. We utilize three widely-used image clustering benchmarking datasets: STL10 (Coates et al., 2011), Cifar10 (Krizhevsky and Hinton, 2009) and ImageNet10 (Deng et al., 2009).

Background Datasets. To assess the robustness of our proposed method against background noise, we include Sports10 (Trivedi et al., 2021) and iNaturalist2021 (Grant Van Horn, 2021), two datasets containing high-resolution images of sports scenes in video games and natural environments.

Human Interpretable Datasets. Three datasets focusing on human concepts rather than individual objects are included. LSUN (Yu et al., 2015), showing, e.g., a living room or a kitchen, Human Activity Recognition (HAR) (Nagadia, 2022), containing scenes such as running and Facial Expression Recognition (FER2013) (Barsoum et al., 2016), e.g., surprise, are considered.

[2] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

4 Text-Guided Image Clustering

We explore the potential of generated text for image clustering. First, we use standard image captioning and observe that the text representation outperforms the image representation of several models. Second, we guide the text generation using VQA models to generate keywords, which we call *keyword-guided clustering*, and introduce *prompt-guided clustering*, where we use domain-specific prompts to elicit relevant properties. Third, we use the generated text for cluster explainability, obtaining keyword-based descriptions for each cluster.

4.1 Caption-Guided Image Clustering

Modern foundation models provide the possibility to work with multiple modalities. In particular, image captioning models describe images with text. Thus, as a first experiment, we investigate how well text clustering on captioned text works in comparison to image clustering, and establish a consistent experimental setup.

Setup. The commonality between current image captioning models is that they consist of an image encoder and a generative LLM to generate text conditioned on the latent image space. As described in Section 3.2 we assess the quality of image and generated text by comparing the clustering performance of the vision encoder embeddings with TF-IDF and SBERT representations using K-Means. We benchmark three SOTA image-to-text models, namely a community-trained version of Flamingo[3] (Alayrac et al., 2022), GIT[4] (Wang et al., 2022a), and BLIP-2[5] (Li et al., 2023), all available within the Huggingface Transformers library (Wolf et al., 2020). Note that we abstain from including dedicated clustering methods (Cai et al., 2022; Qian, 2023; Gao et al., 2021) because they are based on a much weaker image encoder, thus achieving much lower performance. Furthermore, it is not straightforward to train transformer-based image models using clustering objectives. We probabilistically sample a maximum of 80 tokens, without any additional parameters. Only for Flamingo, we set top-K to 8, following the original repository. Experiments were performed on a single A100 40GB and took about 40h hours.

[3] https://huggingface.co/dhansmair/flamingo-mini

[4] https://huggingface.co/microsoft/git-large

[5] https://huggingface.co/Salesforce/blip2-flan-t5-xl

Model	Representation	Standard						Background						Human					
		STL10		Cifar10		ImageNet10		Sports10		iNaturalist2021		FER2013		LSUN		HAR		Avg	
		Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI
Flamingo	Image	95.0	**95.13**	84.0	84.19	**99.38**	**98.85**	**75.87**	**81.61**	40.8	58.09	**36.79**	**17.33**	60.67	60.98	50.07	43.67	67.82	**67.48**
	TF-IDF	82.22	77.0	81.85	76.23	94.32	89.57	54.16	49.86	34.27	43.63	25.77	2.91	**70.58**	64.04	40.92	35.52	60.51	54.85
	SBERT	**97.74**	94.68	**93.64**	**86.15**	98.36	96.05	60.32	55.89	**44.93**	**58.99**	29.79	9.77	68.96	**68.41**	**51.37**	**46.84**	**68.14**	64.6
GIT	Image	51.15	63.62	66.37	64.87	95.41	**93.78**	71.17	75.69	42.47	53.0	24.1	**2.15**	52.06	51.78	38.81	33.18	55.19	54.76
	TF-IDF	79.92	74.71	74.0	66.73	82.69	76.78	**87.42**	84.6	36.12	42.84	25.24	1.66	65.34	57.68	42.87	36.05	61.7	55.13
	SBERT	**96.58**	**93.34**	**86.79**	**76.97**	**96.37**	92.72	85.73	**88.14**	**46.04**	**58.78**	**26.61**	1.95	**69.82**	**61.95**	**48.11**	**42.66**	**69.51**	**64.56**
BLIP-2 (*)	Image	**99.65**	**99.16**	**98.69**	**97.59**	**99.8**	**99.35**	91.31	93.22	44.97	**62.7**	35.97	**21.2**	62.07	64.47	**52.65**	**47.06**	73.14	73.09
	TF-IDF	83.3	79.35	89.0	84.75	93.54	88.81	**99.38**	**98.65**	34.17	39.07	31.86	6.89	76.69	71.05	50.51	46.09	69.81	64.33
	SBERT	98.03	96.27	97.31	94.07	98.22	96.63	99.07	98.47	**47.43**	61.63	**38.21**	20.53	**81.11**	**74.37**	50.85	46.68	**76.28**	**73.58**

Table 1: Comparison of Clustering Accuracy and NMI of image space and generated captions, using TF-IDF and SBERT representations, of multiple Image-to-Text models. For each combination of dataset and metric, underlined numbers represent the best overall performance, and bold numbers the best performance per model. (*) Note that BLIP-2 is pre-trained on ImageNet21K (Deng et al., 2009), which STL10 and ImageNet10 are subsets of.

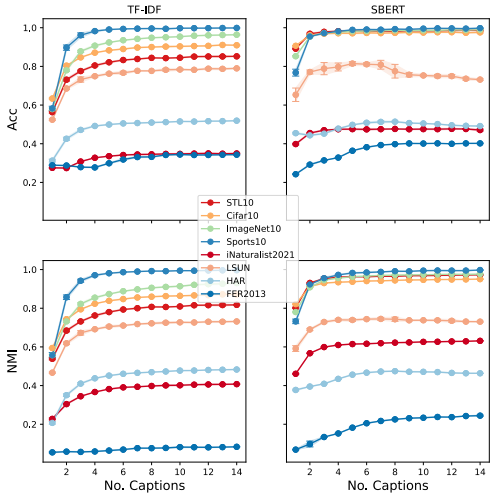


Figure 3: Effect of the number of captions sampled per image for BLIP-2. The number of captions is depicted on the X-axis, mean and standard deviation of clustering performance are on the Y-axis.

We start by studying the effect of the number of captions generated per image. For each amount of captions, we sample 6 versions and report the mean and standard error in Figure 3.

Results. We observe that, for TF-IDF, with a growing number of captions, the performance increases monotonically, whereas SBERT saturates for many datasets. Being counting-based, we think that the reason is that TF-IDF is better at reducing the effect of outlier captions, i.e. single bad captions. For all following experiments, we choose to sample 6 text generations as a trade-off between sampling efficiency and clustering performance.

The full image captioning results are shown in Table 1. The average scores (Avg) show that SBERT outperforms the other two representations across all model types on almost all datasets, while the TF-IDF representation performs worst. Note that we abstain from sophisticated preprocessing such as lemmatization or stemming, which is common for frequency-based representations such as TF-IDF, to keep the setup simple and depend on text information as purely as possible. This might (to a certain degree) explain the worse performance.

Further, we observe that BLIP-2 is the best-performing model. It performs especially well on the standard datasets, which we think is due to the fact that it was pre-trained on ImageNet21k in a self-supervised fashion.

In summary, the results show that text representations, obtained only based on (latent) image representations, provide competitive clustering performance, often outperforming the corresponding image representation.

4.2 Knowledge Injection

Now we investigate the potential of guiding the text generation so that it is specifically suited for clustering. By using modern VQA models, it is possible to elicit dedicated information from images. In the following, we introduce two ways to make use of VQA models.

Keyword-Guided Clustering. Given that it is common to (verbally) describe clusters using keywords, we hypothesize that it is beneficial to prompt the model to generate keywords. The reasons are: 1) keywords provide useful inputs for simpler, traditional count-based representations such as TF-IDF, 2) keywords are useful for count-based analysis methods, such as the proposed cluster explainability algorithm in section 4.3, and 3) ground truth cluster labels (as given by classification datasets used in the clustering literature) are typically described using only a few keywords.

		Sports10		iNaturalist2021		FER2013		LSUN		HAR		Avg	
		Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI	Acc	NMI
Image	ViT	91.31	93.22	44.97	**62.70**	35.97	21.2	62.07	64.47	52.65	47.06	57.39	57.73
Caption-Guided	TF-IDF	**99.38**	**98.65**	34.17	39.07	31.86	6.89	76.69	71.05	50.51	46.09	58.52	52.35
	SBERT	99.07	98.47	47.43	61.63	38.21	20.53	**81.11**	**74.37**	50.85	46.68	63.33	60.34
Keyword-Guided	TF-IDF	99.08	97.82	42.13	48.25	**47.05**	27.34	76.2	69.28	51.35	45.47	63.16	57.63
	SBERT	96.89	96.87	48.44	59.48	46.44	29.96	70.63	70.82	55.66	50.07	63.61	61.44
Prompt-Guided	TF-IDF	84.83	94.46	38.01	47.61	46.86	34.25	66.4	59.92	52.74	47.96	57.77	56.84
	SBERT	98.70	98.12	**48.57**	62.23	45.60	**36.04**	71.59	63.54	**60.93**	**52.94**	**65.08**	**62.57**

Table 2: Comparison of clustering performance of the BLIP-2 image encoder features, and examined types of generated text. For prompt-guided clustering, the clusterings belonging to the prompt with the lowest K-Means are evaluated. For each dataset and metric combination, the best performance is bold, and the second-best performance is underlined.

Prompt-Guided Clustering. In real-world scenarios, often, some domain knowledge about the given data is available. The ability of VQA models to retrieve dedicated information from images opens up the possibility of using domain knowledge in the natural form of text. An example is to ask "Which activity is performed in the picture?". Note, crucially, that this is not possible using standard image clustering models.

Setup. Due to resource constraints, we only use the best-performing (cf. Table 1) image-to-text model, BLIP-2, for the subsequent experiments. Based on the results depicted in Figure 3, we sample $k = 6$ texts for each image.

For keyword-guided clustering, we use the question "Which keywords describe the image?". To perform prompt-guided clustering, we create four questions for each of the datasets. The questions were created by naively transforming the name of the dataset into a question, e.g. for human action recognition "Which activity is performed?" is asked. Note, that no additional prompt engineering efforts were made, as we are not aware of a more principled way to design such prompts. Find all questions in Table 8 in Appendix B.

BLIP-2 solves the "standard" datasets with almost 100% and they exhibit only a collection of objects, making it difficult to pose interesting questions other than 'What objects are described?'. Thus, they are excluded in the following experiments. It is well known that current LLMs possibly generate very different texts, even though the prompt has the same meaning (Elazar et al., 2021). Therefore, in Table 2 we use an unsupervised heuristic to decide which prompt works best by taking the prompt belonging to the clustering with the lowest K-Means loss.

Modality / Question	SBERT	
	Acc	NMI
Image	52.65	47.06
Which keywords describe the image?	55.66	50.07
What type of motion is depicted in the picture?	49.20	42.54
Which activity is shown in the picture?	56.03	49.69
Which action is shown in the picture?	58.68	52.86
What is the person doing in the picture?	**60.93**	**52.94**

Table 3: A case study for prompt-guided image clustering on Human Action Recognition, using the SBERT representation. Find the full table in Appendix B.

Results. In Table 2 we observe that the average performance (Avg) for caption-guided image clustering and SBERT-based keyword-guided clustering is similar. Using keywords, TF-IDF improves on average by 5% for both cluster accuracy and NMI, closing the gap to SBERT. This result is in line with our hypothesis that keywords are a useful representation for image clustering.

As a case study, Table 3 holds the results for the HAR dataset. We observe a notable variance in the performance of multiple prompts. This is a common phenomenon for prompting-based methods (Zhao et al., 2021). Using the K-Means loss as a proxy for selecting the best prompt leads to the best average performance in Table 2.

Interestingly, the confusion matrices in Figure 4 show different assignment patterns depending on the question posed to the VQA model. For instance, when posing the question 'What room is shown in the picture?', all room clusters are formed well, but the others, e.g. bridge or tower, are worse. We argue that this variation is not an issue but a feature of prompt-guided image clustering, e.g., during exploratory data analysis, where one might want to investigate different aspects of a dataset.

In summary, we demonstrate that it is possible

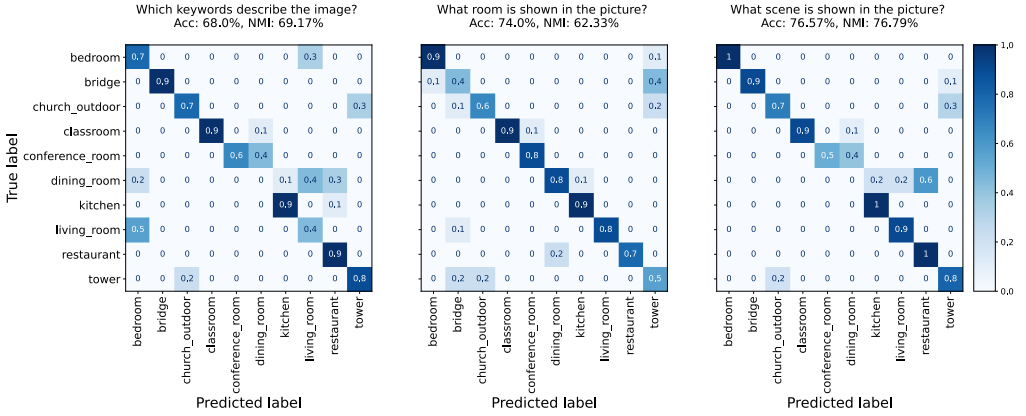


Figure 4: Confusion matrices based on three clustering results from text generated with three different VQA prompts. While a similar cluster accuracy is achieved, we observe that the clustering relates to the prompt. In the middle all room clusters are clustered well, on the right side the clustering is not able to distinguish well between dining room, kitchen and restaurant (see corresponding dining room row), but leads to better overall accuracy.

to improve clustering performance by injecting domain knowledge in the form of text and that the clustering changes according to the posed questions. Further examples of the impact of different prompts on the embedded space and clustering are shown in t-SNE embeddings in Figures 6 and 7 in the Appendix.

4.3 Cluster Explainability

So far, we use the generated text solely to form clusters. But given the (built-in) interpretability of text, a natural extension is to use text as an explanation of the formed clusters. Explainability for image clustering is an important issue, as it provides insights into how the clustering algorithm groups the images, helping users understand the underlying patterns and relationships. The availability of textual descriptions for each cluster sample allows us to extrapolate to textual descriptions of each cluster as a whole.Note that this is not possible using models considering only images.

We hypothesize that a concise way to describe a cluster is to use a small set of keywords. This is based on the fact that the considered datasets use keyword-based labels. Thus, we introduce the following algorithm to obtain keywords for each cluster from the generated text.

Explainability Algorithm. For each predicted cluster, the keywords are sorted by their number of occurrences in the generated texts. The algorithm returns the most frequent keywords per cluster. If a keyword occurs in multiple cluster descriptions, it is not considered, and the next most occurring is chosen. We take the two most occurring keywords based on an initial screening of the LSUN dataset. Find the Pseudocode in the Appendix C.

Setup. We provide a quantitative analysis of the generated descriptions by applying two metrics. First, we introduce the subset exact match (SEM) metric, for which we lowercase each string and check whether the ground truth cluster name appears in the predicted keywords. No further standardization, such as stemming or lemmatization, is performed. Second, SBERT embeddings are used to check the similarity between cluster names and keywords obtained by the explainability algorithm. According to our initial investigation, we use a cosine similarity of 0.4 as the threshold to indicate a match between ground truth and explanation. For each dataset, we provide the cluster accuracy and the explainability performance given the ground truth (*Truth*) clustering and the predicted (*Pred*) clustering, corresponding to the cluster accuracy. Out of the 50 conducted K-Means runs, we use the clustering with the lowest K-Means loss for the analysis.

Results. Table 5 depicts the quantitative evaluation of our algorithm. We observe that the SBERT metric is always equal to or higher than the SEM metric, which makes sense as SEM is a rather strict metric, not understanding synonyms or syntactical changes, e.g., "TableTennis" vs. "table tennis". Interestingly, in most cases, the SBERT metric is higher than the clustering accuracy. Table 4 shows

Ground Truth	Explanation	SEM	SBERT Sim.
Sports10			
AmericanFootball	football, nfl	0	1
Basketball	basketball, basketball game	1	1
BikeRacing	motorcycle, rider	0	1
CarRacing	car, speed	0	0
Fighting	fight, boxing	0	1
Hockey	hockey, hockey game	1	1
Soccer	soccer, soccer game	1	1
TableTennis	ping pong, table tennis	0	0
Tennis	tennis, tennis game	1	1
Volleyball	volleyball, beach	1	1
LSUN			
bedroom	bedroom, bed	1	1
bridge	bridge, river	1	1
church_outdoor	church, cathedral	0	1
classroom	classroom, teacher	1	1
conference_room	meeting, conference	0	1
dining_room	dining room, dining table	1	1
kitchen	kitchen, wood	1	1
living_room	living room, living	1	1
restaurant	restaurant, bar	1	1
tower	tower, city	1	1

Table 4: Examples of generated explanations for Sports10 and LSUN. If a value in the SEM or SBERT Sim. column is 1, it means that the metric says ground truth and explanation match.

an example of generated descriptions and metrics. We observe that both metrics cannot understand that "TableTennis" and "ping pong, table tennis" have the same meaning, but still, all cluster descriptions of Sports10 are correct. For iNaturalist2021 and FER2013, we observe that the generated text is often of bad quality, resulting in low-quality descriptions. We conclude that the generated descriptions provide a good overview of the content of the generated clusters and in most cases, describe the dataset better than clustering accuracy suggests.

5 Broader Impact

We believe there is a lot of unused potential for text as an abstraction in image clustering.

Text as a proxy for "meaningful" clustering. Clustering research aims to find meaningful clusters. In general, it is an open question to define what meaningful exactly stands for, some researchers even call it an ill-posed problem. We argue that text is a good proxy to express meaningfulness as it is based on the natural human form of communication. This is a novel viewpoint on the task of image clustering aligning with research methodologies in the clustering community, where clustering methods are commonly benchmarked with datasets that have human-annotated textual labels as ground truth. Our research contributes to the discussion about meaningful clustering by showing that generated text improves the interpretability of the detected clusters.

	Cluster Acc		SEM		SBERT Sim.	
	TF-IDF	SBERT	Truth	Pred	Truth	Pred
STL10	87	98	100	**100**	100	**100**
ImageNet10	94	99	30	30	100	**100**
CIFAR10	91	97	90	90	100	**100**
Sports10	99	98	50	50	80	80
iNaturalist2021	40	48	0	0	91	45
LSUN	75	68	70	**80**	100	**100**
HAR	51	56	20	13	87	**87**
FER2013	46	46	12	12	38	25

Table 5: Evaluation of our explainability method. In "Truth", the explainability method is applied to the ground truth clustering whereas in "Pred" it is applied to the clustering of the given clustering accuracy. Numbers are boldened if the explainability score of a found clustering ("Pred" columns) outperforms clustering accuracies.

Knowledge Injection. Furthermore, it can be highly subjective what determines a meaningful clustering. For a given dataset, different people are interested in different types of information. For example, in real-world scenarios, an expert might have several questions about a dataset based on their domain knowledge. We show that these questions can be used to guide the clustering process by prompting VQA models. Given the current speed of research, we believe that the increasing ability to use more detailed prompts will drastically improve our knowledge injection method. This, in turn, will open up new research avenues for injecting knowledge into the clustering process.

6 Conclusion

In this work, we introduce *Text-Guided Image Clustering*, using image-captioning and VQA models to automatically generate text, and subsequently cluster only the generated text. After applying multiple captioning models on eight diverse datasets, our experiments show that representations of generated text outperform image representations on many datasets. Further, we use text to include task- and domain knowledge by prompting VQA models, resulting in additional improvements in clustering performance. We find that it is possible to shape the clustering favorably according to the information given by a specific prompt. Additionally, we use the generated text to obtain a keyword-based description for each cluster and show their usefulness quantitatively and qualitatively.

While it is difficult to identify background noise or

irrelevant features in the pixel space, text is discrete and interpretable. We show that text-guided image clustering often outperforms clustering purely on image information, and provides interpretability. Therefore, our research provides insights into the role of text in determining meaningful clusterings.

7 Limitations

While our proposed approach shows promising results, several limitations apply.

Text-guided image clustering is dependent on the quality and effectiveness of the generated text. In cases where the generated text is incomplete, misleading, or fails to capture the essential features of the images, the clustering algorithm may struggle to accurately group similar images. Current image-to-text models are mostly trained on data obtained from the internet. For example, because of licensing and other restrictions, many domain-specific images are not represented appropriately in the training data, resulting in poor text generation abilities for those domains. Nevertheless, our experiments are performed on a wide variety of datasets, more diverse than in common image clustering research, proving the general applicability of the method.

While we show that our approach is effective for image clustering, we do not include results for other visual modalities, such as video or 3D point clouds. We show that it is worthwhile to investigate the possibility of clustering images using generated text and generating textual cluster explanations. The rapid advancement of machine learning models will also enable the same approach for other modalities.

The approach of prompt-guided image clustering is based on the assumption that domain knowledge is readily accessible, allowing the generation of specific questions to guide VQA models. While we show that leveraging domain knowledge can prove advantageous, clustering methods are frequently employed for exploratory data analysis. Introducing domain knowledge may limit the discovery of novel insights or alternative interpretations due to biased prompts.

Acknowledgements

We thank the anonymous reviewers for their constructive feedback. This research has been funded by the Vienna Science and Technology Fund (WWTF)[10.47379/VRG19008] "Knowledge-infused Deep Learning for Natural Language Processing". We thank the European High Performance Computing initiative for providing the computational resources that enabled this work. EHPC-DEV-2022D10-051, EHPC-DEV-2023D11-017.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Emad Barsoum, Cha Zhang, Cristian Canton Ferrer, and Zhengyou Zhang. 2016. Training deep networks for facial expression recognition with crowd-sourced label distribution. In *ACM International Conference on Multimodal Interaction (ICMI)*.

Jinyu Cai, Jicong Fan, Wenzhong Guo, Shiping Wang, Yunhe Zhang, and Zhao Zhang. 2022. Efficient deep embedded subspace clustering. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–30.

Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. 2018. Deep clustering for unsupervised learning of visual features. In *ECCV (14)*, volume 11218 of *Lecture Notes in Computer Science*, pages 139–156. Springer.

Patrick Cavanagh. 2021. The language of vision. *Perception*, 50(3):195–215.

Chandramani Chaudhary, Poonam Goyal, Siddhant Tuli, Shuchita Banthia, Navneet Goyal, and Yi-Ping Phoebe Chen. 2019. A novel multimodal clustering framework for images with diverse associated text. *Multimedia Tools and Applications*, 78:17623–17652.

Adam Coates, Andrew Ng, and Honglak Lee. 2011. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA. PMLR.

Michael C Corballis. 2017. Language evolution: a changing perspective. *Trends in cognitive sciences*, 21(4):229–236.

Thi-Bich-Hanh Dao, Chia-Tung Kuo, S. S. Ravi, Christel Vrain, and Ian Davidson. 2018. Descriptive clustering: Ilp and cp formulations with applications. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 1263–1269. International Joint Conferences on Artificial Intelligence Organization.

Ian Davidson, Antoine Gourru, and S Ravi. 2018. The cluster description problem - complexity results, formulations and approximations. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

Banchiamlack Dessalegn and Barbara Landau. 2013. Interaction between language and vision: It's momentary, abstract, and it develops. *Cognition*, 127(3):331–344.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031.

Jerry A Fodor. 1975. *The language of thought*, volume 5. Harvard university press.

Ricardo Fraiman, Badih Ghattas, and Marcela Svarc. 2011. Interpretable clustering using unsupervised binary trees. *Advances in Data Analysis and Classification*, 7:125–145.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

macaodha Grant Van Horn. 2021. inat challenge 2021 - fgvc8.

Philip Häusser, Johannes Plapp, Vladimir Golkov, Elie Aljalbout, and Daniel Cremers. 2018. Associative deep clustering: Training a classification network with no labels. In *GCPR*, volume 11269 of *Lecture Notes in Computer Science*, pages 18–32. Springer.

Ray Jackendoff, Paul Bloom, Mary A Peterson, Lynn Nadel, and Merrill F Garrett. 1996. Language and space. *chapter "The Architecture of the Linguistic-Spatial Interface*, pages 1–30.

Cheng Jin, Wenhui Mao, Ruiqi Zhang, Yuejie Zhang, and Xiangyang Xue. 2015. Cross-modal image clustering via canonical correlation analysis. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 151–159. AAAI Press.

Turkay Kart, Wenjia Bai, Ben Glocker, and Daniel Rueckert. 2021. Deepmcat: Large-scale deep clustering for medical image categorization. In *Deep Generative Models, and Data Augmentation, Labelling, and Imperfections*, pages 259–267, Cham. Springer International Publishing.

Alex Krizhevsky and Geoffrey Hinton. 2009. Learning multiple layers of features from tiny images. Technical Report 0, University of Toronto, Toronto, Ontario.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In *ICML*.

Himanshu Mittal, Avinash Pandey, Mukesh Saraswat, Sumit Kumar, Raju Pal, and Garv Modwel. 2021. A comprehensive survey of image segmentation: clustering methods, performance parameters, and benchmark datasets. *Multimedia Tools and Applications*, 81.

Michal Moshkovitz, Sanjoy Dasgupta, Cyrus Rashtchian, and Nave Frost. 2020. Explainable k-means and k-medians clustering. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7055–7065. PMLR.

Meet Nagadia. 2022. Human action recognition (har) dataset.

Chuang Niu, Jun Zhang, Ge Wang, and Jimin Liang. 2020. Gatcluster: Self-supervised gaussian-attention network for image clustering. In *ECCV (25)*, volume 12370 of *Lecture Notes in Computer Science*, pages 735–751. Springer.

Martin A Nowak, Natalia L Komarova, and Partha Niyogi. 2002. Computational and evolutionary aspects of language. *Nature*, 417(6889):611–617.

Shreelekha Pandey and Pritee Khanna. 2016. Content-based image retrieval embedded with agglomerative clustering built on information loss. *Computers & Electrical Engineering*, 54:506–521.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos,

D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Steven Pinker and Paul Bloom. 1990. Natural language and natural selection. *Behavioral and Brain Sciences*, 13(4):707–727.

Qi Qian. 2023. Stable cluster discrimination for deep clustering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 16645–16654.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Prathyush Sambaturu, Aparna Gupta, Ian Davidson, S. S. Ravi, Anil Vullikanti, and Andrew Warren. 2020. Efficient algorithms for generating provably near-optimal cluster descriptors for explainability. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(02):1636–1643.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Chintan Trivedi, Antonios Liapis, and Georgios N Yannakakis. 2021. Contrastive learning of generalized game representations. In *2021 IEEE Conference on Games (CoG)*. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(95):2837–2854.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Computer Vision and Pattern Recognition*.

Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. 2022a. GIT: A generative image-to-text transformer for vision and language. *Transactions on Machine Learning Research*.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022b. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 478–487. JMLR.org.

Jie Xu, Huayi Tang, Yazhou Ren, Liang Peng, Xiaofeng Zhu, and Lifang He. 2022. Multi-level feature learning for contrastive multi-view clustering. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16030–16039.

Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. 2017. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3861–3870. PMLR.

Sean T Yang, Kuan-Hao Huang, and Bill Howe. 2021. Jecl: Joint embedding and cluster learning for image-text pairs. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8344–8351. IEEE.

Yi Yang, Dong Xu, Feiping Nie, Shuicheng Yan, and Yueting Zhuang. 2010. Image clustering using local discriminant models and global integration. *IEEE Trans. Image Process.*, 19(10):2761–2773.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*.

Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel C. F. Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. 2021. Florence: A new foundation model for computer vision. *ArXiv*, abs/2111.11432.

Hongjing Zhang and Ian Davidson. 2021. Deep descriptive clustering. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3342–3348. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Wen Zhang, Taketoshi Yoshida, and Xijin Tang. 2011. A comparative study of tf* idf, lsi and multi-words for text classification. *Expert Systems with Applications*, 38(3):2758–2765.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12697–12706. PMLR.

Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan L. Yuille, and Tao Kong. 2022a. Image BERT pre-training with online tokenizer. In *ICLR*. OpenReview.net.

Sheng Zhou, Hongjia Xu, Zhuonan Zheng, Jiawei Chen, Zhao Li, Jiajun Bu, Jia Wu, Xin Wang, Wenwu Zhu, and Martin Ester. 2022b. A comprehensive survey on deep clustering: Taxonomy, challenges, and future directions. *ArXiv preprint*, abs/2206.07579.

A Dataset Description

Here, we provide some additional information about the datasets. An overview of the datasets is given in Table 6, including name, number of classes, number of images, and size, given in pixels. You can find examples of images of each dataset in Table 7.

In the following, there is a small description of the datasets, including the class labels, provided in their original form which we also use in the evaluation of our explainability algorithm.

STL10 (Coates et al., 2011). This traditional dataset consists of 10 classes, namely "deer, horse, bird, cat, ship, airplane, car, truck, monkey, dog". We use the full dataset, i.e. train and test split. Note, that it is inspired by Cifar10 and attempts to be more complicated because it contains fewer images.

Cifar10 (Krizhevsky and Hinton, 2009). The dataset is comprised of 10 similar object classes: "deer, horse, bird, automobile, airplane, cat, ship, truck, dog, frog". Again, we use the full dataset.

ImageNet10. Imagenet-10 is a subset of the larger ImageNet dataset, containing 10 classes. Given the hierarchical nature of of ImageNet, each class is described by multiple keywords: 'trailer truck, tractor trailer, trucking rig, rig, articulated lorry, semi', 'snow leopard, ounce, Panthera uncia', 'airliner', 'Maltese dog, Maltese terrier, Maltese', 'sports car, sport car', 'orange', 'soccer ball', 'airship, dirigible', 'container ship, containership, container vessel', 'king penguin, Aptenodytes patagonica'

Sports10 (Trivedi et al., 2021). The Sports-10 dataset provides labeled images from 175 video games across 10 sports genres. The labels are "CarRacing, Tennis, AmericanFootball, BikeRacing, TableTennis, Fighting, Basketball, Hockey, Soccer, Volleyball".

Inaturalist2021 (Grant Van Horn, 2021). The full dataset contains images of 10,000 species separated into 10 classes, which are "Animalia, Arachnids, Amphibians, Birds, Insects, Ray-finned Fishes, Plants, Mollusks, Reptiles, Fungi, Mammals". We experiment with the validation set.

Dataset Group	Name	No. of classes	No. of Images	Size (pixels)
Standard	STL10	10	13000	96x96
	ImageNet10	10	13000	500x364
	CIFAR10	10	60000	32x32
Background	Sports10	10	3000	1280x720
	iNaturalist 2021	11	100000	284x222
Human	LSUN	10	3000	341x256
	Human Action Recognition	15	18000	240x160
	FER2013	8	35488	48x48

Table 6: Overview over some basic dataset statistics.

LSUN (Yu et al., 2015). The Large-Scale Scene Understanding (LSUN) dataset offers labeled images depicting scenes from the following categories: "conference_room, dining_room, bedroom, church_outdoor, bridge, tower, restaurant, living_room, classroom, kitchen". We experiment with the test set.

HAR (Nagadia, 2022). contains images of human activities. They are "running, sleeping, listening_to_music, texting, drinking, clapping, fighting, eating, sitting, using_laptop, cycling, calling, laughing, hugging, dancing".

FER2013 (Barsoum et al., 2016). The Facial Expression Recognition 2013 dataset consists of labeled grayscale images depicting human facial expressions, which are "surprise, anger, contempt, happiness, fear, disgust, sadness, neutral".

B Knowledge Injection

In section 4.2 we introduce prompt-guided clustering. For each dataset, multiple prompts are tested. They are generated by adapting the dataset name and transforming them into a question. Table 8 encompasses all prompts used in our experimental setup, accompanied by the corresponding evaluation performance metrics, namely Cluster Accuracy (Acc) and Normalized Mutual Information (NMI) for the image encoder representation, and the TF-IDF and SBERT representations. The used model is BLIP-2. Further, we provide a visual inspection of the same numbers in Figure 5.

In order to get a better understanding of the comparison of embedding structure, and how generated text relates to that, we provide two examples. In Figure 6 there is an example of the LSUN dataset and in Figure 7 there is a corresponding example of the Sports10 dataset.

C Explainability

In this section, we provide pseudo-code for the algorithm in section 4.3. As described previously, it counts the number of keyword occurrences per cluster. Afterwards, it takes the top two exclusive keywords.

Algorithm 1 Explainability

Require:
$X = \{X_1, X_2, ..., X_m\}$: be the set of keyword lists for each sample,
$Y = \{Y_1, Y_2, ..., Y_m\}$: be the set of (predicted) cluster labels for each sample,
n : Number of output keywords per cluster.
Ensure: List
procedure SIMPLEXAI(X, Y)
  A, O ← [], [] ▷ Active keywords, and others
  for i `in unique(`Y`)` **do**
    $K \leftarrow$ count-ordered list of keywords cluster i
    $A[i] \leftarrow K[0:n]$
    $O[i] \leftarrow K[n:]$
  end for
  while $\bigcap_i A[i] \neq \emptyset$ **do** ▷ Remove duplicates
    $D \leftarrow \bigcap_i A[i]$
    $A[i] \leftarrow A[i] \setminus D$
    $A[i] \leftarrow A[i] \cup O[0:|D|]$
    $O[i] \leftarrow O[2|D|:]$
  end while
  return A
end procedure

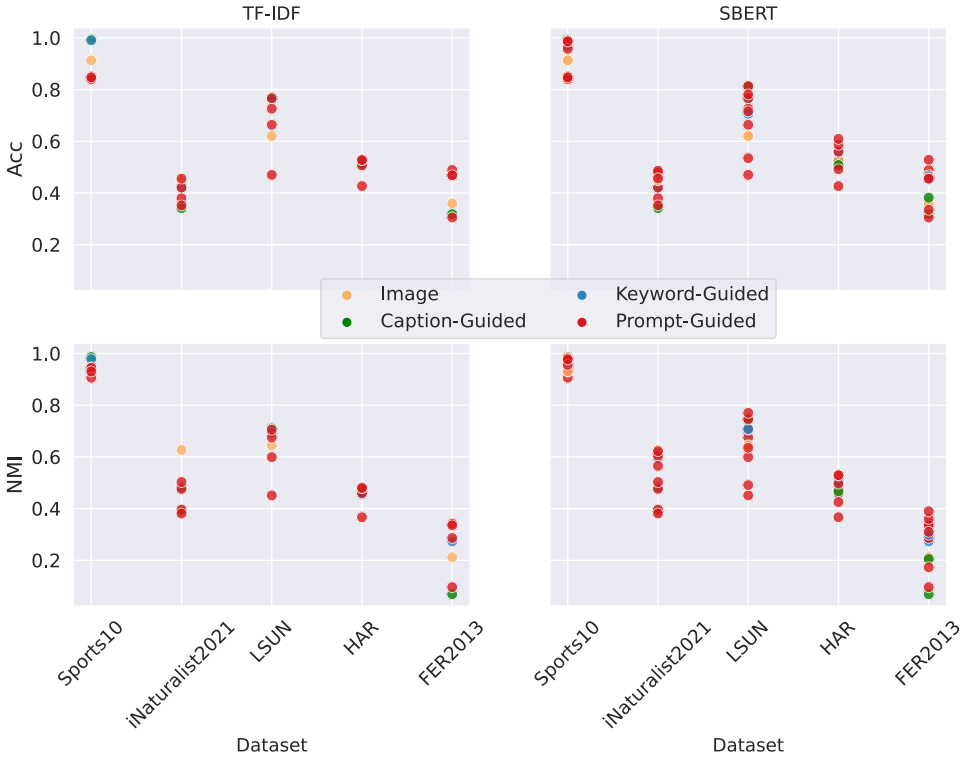


Figure 5: Comparison of all used strategies. Find the questions for prompt-guided clustering in Table 8.

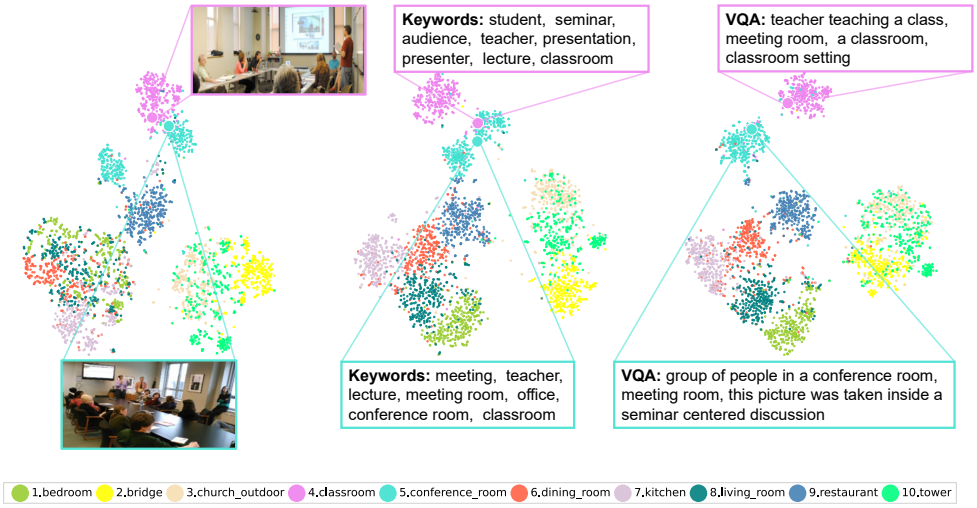


Figure 6: t-SNE embeddings of BLIP2 for the LSUN dataset. From left to right: Image embedding (Acc: 63.11), Keyword SBERT embedding (Acc: 71.12) and VQA SBERT embedding (Acc: 81.83 with prompt: "What environment is shown in the picture?"). The improvement in cluster accuracy corresponds to better separated clusters in the t-SNE embeddings.

Dataset	Image1	Label1	Image2	Label2
STL10		bird		car
CIFAR10		automobile		horse
ImageNet10		airship, dirigible		soccer ball
Sports10		CarRacing		BikeRacing
iNaturalist2021		Birds		Insects
LSUN		kitchen		bridge
Human Action Recognition		cycling		running
FER2013		anger		happiness

Table 7: Examplatory images of the datasets. The images contain different properties, such as image quality or background noise. Also, the labels vary in their syntax and semantic meaning, e.g. objects vs. movements.

Dataset	Modality / Question	Image		TF-IDF		SBERT	
		Acc	NMI	Acc	NMI	Acc	NMI
Sports10	Image	91.31	93.22				
	Caption			99.38	98.65	99.07	98.47
	Keyword			99.08	97.82	96.89	96.87
	Which sport is shown in the picture?			84.89	94.57	98.7	98.12
	What type of sport is shown in the picture?			84.83	94.46	99.0	98.21
	Which game is shown in the picture?			84.0	90.64	95.77	95.58
	Which sports contest is shown in the picture?			84.76	93.06	98.64	97.7
iNaturalist2021	Image	44.97	62.7				
	Caption			34.17	39.07	47.43	61.63
	Keyword			42.13	48.25	48.44	59.48
	What type of biological object is shown in the picture?			38.01	47.61	47.14	61.21
	What is the biological classification of the object in the picture?			35.23	39.66	47.82	60.43
	Which biological category is shown in the picture?			42.1	50.3	48.57	62.23
	Which species is shown in the picture?			45.57	38.13	45.65	56.55
LSUN	Image	62.07	64.47				
	Caption			76.69	71.05	81.11	74.37
	Keyword			76.2	69.28	70.63	70.82
	What location is shown in the picture?			47.04	45.12	53.49	49.11
	What kind of environment is shown in the picture?			72.63	67.52	81.37	74.6
	What room is shown in the picture?			66.4	59.92	71.59	63.54
	What scene is shown in the picture?			76.71	70.5	78.15	77.05
HAR	Image	52.65	47.06				
	Caption			50.51	46.09	50.85	46.68
	Keyword			51.35	45.47	55.66	50.07
	What type of motion is depicted in the picture?			42.68	36.69	49.2	42.54
	Which activity is shown in the picture?			50.77	46.04	56.03	49.69
	Which action is shown in the picture?			52.75	48.13	58.68	52.86
	What is the person doing in the picture?			52.74	47.96	60.93	52.94
FER2013	Image	35.97	21.2				
	Caption			31.86	6.89	38.21	20.53
	Keyword			47.05	27.34	46.44	29.96
	What type of countenance is shown in the picture?			30.53	9.64	33.53	17.34
	Which emotion is shown in the picture?			46.86	34.25	45.6	36.04
	Which facial expression is shown in the picture?			48.93	33.55	52.85	39.0
	Which mood is shown in the picture?			46.89	28.66	45.54	31.03

Table 8: Full evaluation table for all prompts. All representations, image and text are based on the BLIP-2 model.

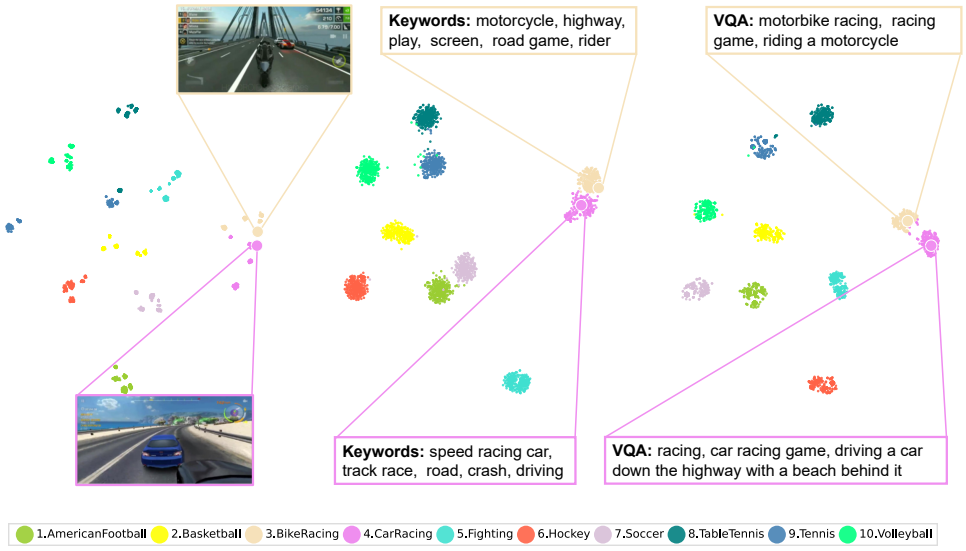


Figure 7: t-SNE embeddings of BLIP2 for the Sports10 dataset. From left to right: Image embedding (Acc: 91.31), Keyword SBERT embedding (Acc: 96.89) and VQA SBERT embedding (Acc: 99.00 with prompt: "What type of sport is shown in the picture?"). The improvement in cluster accuracy corresponds to better separated clusters in the t-SNE embeddings.

8. Text-Guided Alternative Image Clustering

Title	Text-Guided Alternative Image Clustering
Authors	Andreas Stephan, Lukas Miklautz, Collin Leiber, Pedro Henrique Luz De Araujo, Dominik Répás, Claudia Plant and Benjamin Roth
Publication Outlet	In Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024), pages 177–190, Bangkok, Thailand. Association for Computational Linguistics.
Copyright Information	
Thesis-Reference	Stephan et al. (2024a)

Supplementary Material

The code is available on Github at https://github.com/AndSt/alternative_image_clustering.

Author Contributions

Andreas Stephan	developed and expanded on the initial idea. He implemented most of the code, designed the experiments, conducted the experiments, analyzed them, and iterated on them. Further, he wrote the majority of the paper.
Lukas Miklautz	conceived the initial idea, participated in developing and expanding the initial idea, provided feedback on the modeling and experiments, and helped write the paper.
Collin Leiber	provided feedback on the modeling, experimental design, and experiments, implemented a baseline, and helped write the paper.
Pedro Henrique Luz De Araujo	provided feedback on the modeling and experiments, and helped write the paper.
Dominik Répás	wrote an initial implementation, conducted initial experiments, and helped write the paper.
Claudia Plant	provided feedback on the initial idea, on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.
Benjamin Roth	provided feedback on the initial idea, on the modeling, the experimental design, and experiments, revised the draft, and in general, provided mentoring and supervision.

Text-Guided Alternative Image Clustering

Andreas Stephan[1,2,7], **Lukas Miklautz**[1], **Collin Leiber**[5,6], **Pedro Henrique Luz de Araujo**[1,2], **Dominik Répás**[1], **Claudia Plant**[1,3] and **Benjamin Roth**[1,3,4]
[1] Faculty of Computer Science, [2] UniVie Doctoral School Computer Science,
[3] ds:UniVie, [4] Faculty of Philological and Cultural Studies,
University of Vienna, Vienna, Austria
[5] LMU Munich, Germany
[6] Munich Center for Machine Learning, Munich, Germany
[7] andreas.stephan@univie.ac.at

Abstract

Traditional image clustering techniques only find a single grouping within visual data. In particular, they do not provide a possibility to explicitly define multiple types of clustering. This work explores the potential of large vision-language models to facilitate alternative image clustering. We propose Text-Guided Alternative Image Consensus Clustering (TGAICC), a novel approach that leverages user-specified interests via prompts to guide the discovery of diverse clusterings. To achieve this, it generates a clustering for each prompt, groups them using hierarchical clustering, and then aggregates them using consensus clustering. TGAICC outperforms image- and text-based baselines on four alternative image clustering benchmark datasets. Furthermore, using count-based word statistics, we are able to obtain text-based explanations of the alternative clusterings. In conclusion, our research illustrates how contemporary large vision-language models can transform explanatory data analysis, enabling the generation of insightful, customizable, and diverse image clusterings. [1]

1 Introduction

Exploratory data analysis (EDA) is crucial in the comprehension and analysis of data (Tukey, 1970). Clustering arises as a cornerstone EDA methodology, facilitating the grouping of similar data objects into coherent groups. A dataset of images, for example, can be clustered based on semantic similarities between the shown objects. Nevertheless, within applied contexts, variations in user requirements or foci demand distinct clustering formations. One might, for instance, cluster a dataset of cards by rank or by suit (see Figure 1). In such circumstances, it is advantageous to derive multifaceted insights into a dataset from diverse perspectives.

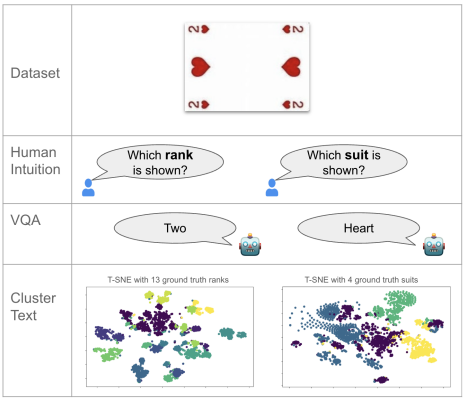


Figure 1: Assume we have an image of a card depicting a "heart two". Given two different user queries, the VQA model gives different responses. Clustering the generated text based on different prompts results in alternative clusterings that satisfy different needs. The colors in the figure represent the ground truths of "rank" and "suit" for different generated texts.

Current approaches in alternative image clustering either rely on image-based features (Mautz et al., 2018; Miklautz et al., 2020) or utilize text through image-text bi-encoders, often with architectures resembling CLIP (Radford et al., 2021; Yao et al., 2024). These methods, while powerful, neglect the rich insights that can be extracted by models explicitly trained to retrieve specific aspects of information from images using text (e.g., visual question answering (VQA) models (Antol et al., 2015)). Stephan et al. (2024) demonstrate the effectiveness of using generated text descriptions to improve standard image clustering tasks, i.e. in scenarios where a single clustering structure is expected.

We aim to use image-to-text models to obtain alternative clusterings. By encoding visual content into text, similarity dimensions beyond the visual features can be explored, potentially revealing interpretable relationships. We introduce *TGAICC*

[1] Code available at https://github.com/AndSt/alternative_image_clustering.

Proceedings of the 9th Workshop on Representation Learning for NLP (RepL4NLP-2024), pages 177–190

(Text-Guided Alternative Image Consensus Clustering), a clustering method that uses the output of multiple image-to-text models to obtain alternative clusterings. TGAICC incorporates VQA models to generate multiple textual descriptions of images and then clusters the images based on the generated natural language descriptions. We identify similar clusterings using their mutual information, group them using hierarchical clustering and aggregate them using consensus clustering to form refined, alternative clusterings.

Our experimental setup employs four widely used alternative image clustering datasets, each possessing two or three ground truth labelings (e.g., playing cards clustered by rank or suit). We compare TGAICC against baselines for alternative clustering using image-only features and baselines that make use of the generated text. Our experiments demonstrate the following key findings: methods clustering the generated text outperform methods based on image features on these alternative clustering datasets, underscoring the power of textual representations in capturing diverse aspects of similarity. Further, TGAICC, on average, achieves superior results across the evaluated datasets and metrics when compared to all other methods, highlighting the effectiveness of our framework in leveraging image-to-text models to uncover alternative and insightful clusterings. Lastly, we can better interpret the clusterings by explaining the content using word statistics. Our case study on the cards dataset shows that text provides an opportunity to obtain an informative overview of the data.

In summary, our research provides the following contributions:

1. We introduce a prompt-based setup to obtain alternative image clusterings.

2. We introduce TGAICC, a method that combines ideas from multi-modality, hierarchical clustering, and consensus clustering to obtain alternative clusterings.

3. Our experiments on four common alternative image clustering datasets show that TGAICC outperforms baseline algorithms.

4. Our methodology enables the ability to generate textual cluster explanations, offering a clear overview of the unique content captured within each alternative clustering.

2 Related Work

This work builds upon image clustering, consensus clustering, and alternative clustering approaches. We provide a brief overview of these relevant areas and describe the necessary background.

2.1 Image Clustering

Research in image clustering has addressed several standard issues, and a variety of techniques have been developed to tackle them. (Ezugwu et al., 2022) provide a survey on clustering approaches. Classic approaches like k-means (Lloyd, 1982) have demonstrated effectiveness but often struggle with complex or high-dimensional image data. To address these limitations, more recent work has explored deep clustering methods such as DEC (Xie et al., 2016) and IDEC (Guo et al., 2017). In addition to these core techniques, representation learning and more specifically, self-supervised learning (Jaiswal et al., 2021) has emerged as a vital aspect of image clustering (Lehner et al., 2023; Adaloglou et al., 2023). In Contrastive Clustering (Li et al., 2021), the authors use one loss contrasting image features and another loss contrasting clustering features, i.e., the predicted cluster of two augmentations of the same image. A different approach is used in Text-Guided Image Clustering (Stephan et al., 2024). This paradigm leverages image-to-text models and subsequently cluster text. The observation that text often outperforms image-based features motivates this work.

2.2 Consensus Clustering

Variability in clustering results arises from different clustering algorithms or variations in their initializations. Given that different clusterings potentially reveal different insights (e.g., accurately identifying a cluster representing "hearts"). Consensus clustering methods aim to aggregate results from multiple base clustering algorithms to produce a more robust and stable final clustering. The problem was formalized by (Strehl and Ghosh, 2002) and the authors introduce the Cluster-based Similarity Partitioning Algorithm (CSPA), HyperGraph Partitioning Algorithm (HGPA), and the Meta-CLustering Algorithm (MCLA). All three methods employ similarity functions, e.g. Normalized Mutual Information (NMI), to construct a similarity graph and use graph theory to obtain a consensus clustering. In (Li and Ding, 2008), non-negative matrix factorization (NMF) is used to obtain a consensus clustering.

The Hybrid Bipartite Graph Formulation (HBGF) (Fern and Brodley, 2004) employs a bipartite graph representation. In (Miklautz et al., 2022), the authors introduce DECCS, a deep learning-based consensus method, which learns a representation on which heterogeneous clustering algorithms share a consensus on the obtained clusterings.

2.3 Alternative Clustering

Clustering methods usually focus on finding a single optimal clustering solution. Motivated by the fact that there may be multiple meaningful ways to group data points, alternative clustering approaches aim to uncover multiple, diverse clustering structures within the same data (Yu et al., 2024; Müller et al., 2012).

Cui et al. (2007) first apply a traditional clustering algorithm and then transform the dataset into a feature space orthogonal to the current clustering. Two strategies are proposed: orthogonal clustering (orth1) and clustering in orthogonal subspaces (orth2). In contrast, Non-redundant K-means (Nr-Kmeans) (Mautz et al., 2018) simultaneously identifies multiple clusterings within a dataset by iteratively rotating the feature space and assigning features to specific clusterings. ENRC (Miklautz et al., 2020) is a deep non-redundant clustering method that learns multiple clusterings from a dataset by (soft-)assigning each dimension of the embedded space to a clustering. In (Kwon et al., 2024), the authors provide initial text criteria, e.g., suits and ranks, and use image-to-text models to extract information, and then GPT-4 to obtain cluster names and classify images into clusters. Thus, this approach is expensive. In concurrent work, (Yao et al., 2024) use GPT-4 to generate cluster name candidates and contrastively fine-tune CLIP (Radford et al., 2021).

2.4 Image-To-Text Models

Recently, the development of multimodal models has seen rapid advancement. Image-to-text models, in particular, learn to associate visual content with corresponding textual descriptions, which is useful for, e.g., visual question-answering (VQA) (Yin et al., 2023; Antol et al., 2015).

Flamingo (Alayrac et al., 2022) allows interleaving images and text by using Perceiver Resamplers on top of pre-trained models. BLIP and BLIP2 (Li et al., 2022, 2023a) employ a frozen image encoder along with a frozen LLM to generate text. LLaVA and LLaVA-NeXT(Liu et al., 2023b,a) convert image patches into token embeddings using a fixed Vision Transformer encoder followed by a trained MLP. These tokens then become the input for the LLM, enhancing the descriptive results.

In this work, we use LLaVA to extract relevant information from images. More specifically, we frame the image-to-text generation as a VQA task: we prompt LLaVA with an image and corresponding questions about it to generate natural language descriptions of the image.

3 TGAICC

We use image-to-text models, specifically models that are able to describe specific aspects of information from images in order to obtain different clusterings. Thus, we design prompts to perform VQA. It is well known (Bach et al., 2022; Sclar et al., 2024) that responses to seemingly semantically equal prompts might vary heavily. Thus, we use multiple formulations of each prompt and aggregate their clusterings afterward. Figure 2 gives an overview of the process.

Setup. The input to TGAICC is a dataset of k datapoints, t initial prompts, and, as common in the alternative-clustering literature, the number of clusters in the ground truth clusterings $\{z_1, \ldots, z_t\}, z_i \in \mathbb{N}$. E.g., $\{2, 4\}$ means the algorithm should return one clustering with 2 and one clustering with 4 clusters. Note that the difference between the traditional alternative clustering setup and ours is that we assume additional initial prompts. The output is comprised of t clusterings where the k data points are grouped into $z_1, \ldots, z_t$ clusters.

3.1 Initialization

The initialization encompasses step 1 to step 4 in Figure 2 and returns a set of clusterings.

Prompt Design In Step 1, we write a query and ask GPT-4 (OpenAI et al., 2024) to automatically generate additional questions. The specific prompt is '*Generate three diverse paraphrases for the following question: {initial question}*'. Further, we generate a variation of each prompt by appending the directive "Write concisely.", aiming to reduce the verbosity of the responses. The output is depicted in Step 2. This is based on the observation that image clusters are often described by succinct short descriptors, e.g., the datasets in our experiments or ImageNet-based (Deng et al., 2009) clustering datasets. Thus, these prompts align with our knowledge about the clustering tasks.

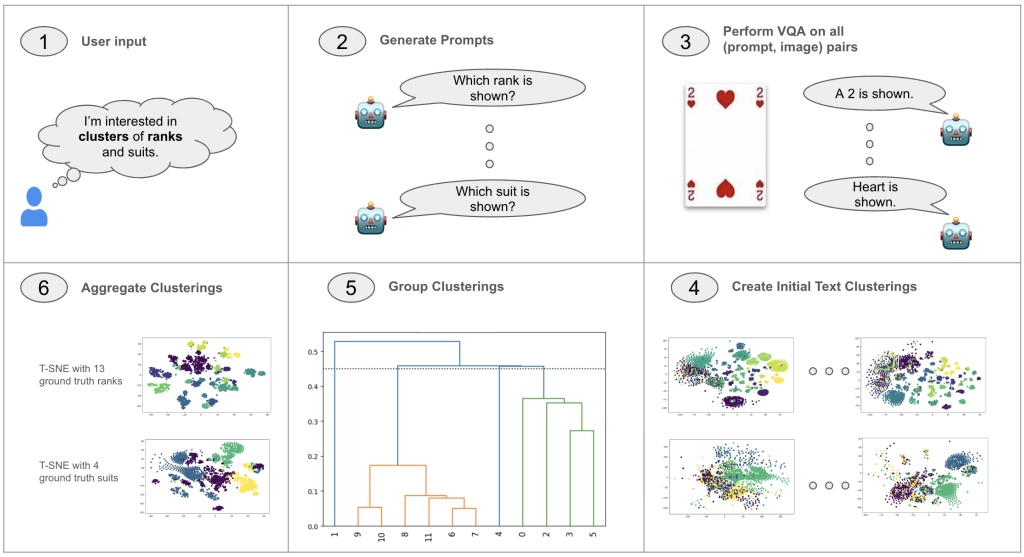


Figure 2: An overview of our methodology. In 1) a user provides text, indicating his interest in the data. In 2) a LLM generates a set of prompts tailored to extract specific information from images, and in 3) VQA is performed for each prompt on each data sample. In 4) the texts generated per prompt are clustered (colors resemble ground truth "rank" and "suit"). In 5) a hierarchy of similar clusterings is built. Based on a threshold (dotted line), multiple groups of clusterings (green and orange) are identified and in 6) aggregated to obtain the final alternative clusterings.

Initial Clustering In Step 3, we perform VQA for each pair of images and prompts, generating responses relevant to the visual content. Next, in Step 4, we create text representations using both traditional TF-IDF (Sparck Jones, 1988) and an advanced sentence embedding model, namely *gte-large* (Li et al., 2023b). Finally, we apply k-means to these text representations and obtain a clustering for each prompt and each representation.

3.2 Grouping

The input to the grouping stage are n pairs of prompt and corresponding clustering $(p_i, \pi_i), i \in [n]$ and the number of ground truth clustering sizes $\{z_1, \ldots, z_t\}, z_i \in \mathbb{N}$, e.g. $\{2, 4\}$. The goal is to obtain groups of clusterings to later find consensus between the individual clusterings explaining the data from a similar perspective. This is displayed in Step 5 of Figure 2. Specifically, we aim to connect semantically similar clusterings and detect potential outlier clusterings, which are caused by prompts leading to unexpected or inconsistent VQA outcomes and are not useful for our final clustering. Find examples of generated text in Table 6.

Therefore, we compute the similarity of two clusterings using Adjusted Mutual Information (AMI) (Vinh et al., 2010). We choose AMI as it is a standard clustering metric based on information theory. Then, we use a spanning-tree-based hierarchical clustering (Müllner, 2011) algorithm[2] to systematically group similar prompts, facilitating a structured analysis of clustering behavior (see Step 5 of Figure 2). The basic idea is that for a threshold $\tau \in (0, 1)$, two clusterings A, B are connected if their distance is less than τ, i.e. $AMI(A, B) < \tau$. Here, we use two strategies, which we call "min" and "max". For "min", we find a minimum threshold such that the resulting number of groups is equal to the number of expected groupings t. For "max", we find a maximum threshold such that this constraint is fulfilled. We use the trivial solution to iterate over all thresholds in $\tau \in (0, 1)$ in steps of 0.02 as the runtime is negligible.

3.3 Aggregation

In the end, we synthesize each group of clusterings. Given that we aggregate potentially very different clusterings, it is beneficial to use different aggregation schemes. Therefore, we apply multi-

[2]Algorithm is readily available in the Scipy library (Virtanen et al., 2020): https://docs.scipy.org/doc/scipy/reference/generated/scipy.cluster.hierarchy.fcluster.html

ple consensus clustering algorithms for each group and choose the instance with the lowest clustering loss. Specifically, we employ MCLA, HBGF, CSPA, and NMF (Strehl and Ghosh, 2002; Li and Ding, 2008) to aggregate the clusterings within the groups[3]. Thereby, we aim to use consensus clustering to combine the strength of multiple clusterings. In our ablation analysis we also test the simple solution where we concatenate the generated text of the prompts in each clustering group and perform k-means on the concatenated string.

4 Experiments

In this section, we introduce the experimental setup. Afterwards, we discuss the main results obtained, highlighting the performance of TGAICC and text-based methods. Additionally, we provide an ablation study, systematically analyzing the impact of various components and prompts on the overall performance. Finally, we perform a simple cluster explainability method to get a textual overview of the data.

Dataset	#samples	#clusters	Size
Fruits-360	4856	4; 4	100x100
Cards	8029	3; 4	224x224
GTSRB	6720	4; 2	15x15 to 250x250
NR-Objects	10000	6; 2; 3	100x100

Table 1: Overview of statistics of the dataset. The third column contains the number of clusters in the ground truth clusterings.

4.1 Experimental Setup

In this section, we outline the key components of our experimental setup, including the evaluation metrics, data representations, and models used. All experiments were run on a single A100 GPU. VQA took approximately 24 hours, and TGAICC experiments took about the same amount of time. Embedding text and running consensus clustering are the most time-consuming elements. Each algorithm is executed 10 times with different random states, and we report the average performance across these runs.

4.1.1 Metrics

We employ two widely used metrics to assess the performance of our clustering models. The Adjusted Rand Index (ARI) (Rand, 1971) measures the similarity between the predicted cluster assignments and the ground truth labels, adjusting for chance agreement. The Adjusted Mutual Information (AMI) (Vinh et al., 2010) quantifies the shared information between the predicted clusters and the true labels. We multiply by 100 to increase readability.

4.1.2 Representations

We utilize image- and text-based representations to capture different aspects of the data.

Image Embeddings: We utilize the LLaVA-NeXT model (Liu et al., 2023a), which incorporates the image encoder of a frozen CLIP model. This allows us to directly use the image embeddings learned during the contrastive pre-training of CLIP (Radford et al., 2021) for our clustering tasks.

Statistical text embeddings: We employ Term Frequency-Inverse Document Frequency (TF-IDF) embeddings, a standard word frequency-based technique for representing documents.

Neural text embeddings: To better capture semantic relationships, we employ the "gte-large"[4] model (Li et al., 2023b), a state-of-the-art sentence encoder.

4.1.3 Datasets

In the following, we briefly describe the used datasets. Table 1 summarizes the relevant statistics for all datasets. More details about datasets and corresponding prompts are given in Appendix A.

Cards[5] This dataset is primarily used for classification tasks but contains attributes suitable for clustering based on the suit and rank of the cards.

Fruits-360 (Mureşan and Oltean, 2018) The dataset is composed of images that can be clustered by fruit type (citrus, berries, etc.) and color.

NR-Objects (Miklautz et al., 2020) The dataset contains images of objects (e.g., cubes), which can be clustered by shape, material, or color.

German Traffic Sign Recognition (GTSRB) (Houben et al., 2013) This dataset contains traffic signs and can be clustered by color and traffic sign type.

[3]We used the library Cluster Ensembles: https://github.com/GGiecold-zz/Cluster_Ensembles

[4]Model is available on Hugging Face (https://huggingface.co/thenlper/gte-large, and is used via the Sentence-BERT (SBERT) library (Reimers and Gurevych, 2019)

[5]https://www.kaggle.com/datasets/gpiosenka/cards-imagedatasetclassification

Dataset	Type		Image					TF-IDF		SBERT		TGAICC
			k-means	orth-1	orth-2	Nr-Kmeans	ENRC	Avg. Prompt	Concatenate	Avg. Prompt	Concatenate	
Fruits-360	fruit	ARI	27.40	31.50	30.80	**35.40**	26.00	14.80	20.10	15.10	17.20	18.60
		AMI	41.30	42.10	42.90	**50.60**	36.70	24.80	32.20	25.00	28.60	26.90
	colour	ARI	33.20	35.40	33.50	40.90	39.70	40.00	54.60	47.40	51.60	**54.70**
		AMI	47.30	53.30	51.70	55.50	54.90	50.70	**65.50**	56.90	60.80	64.80
GTSRB	type	ARI	41.70	46.80	46.80	22.70	38.20	45.10	**61.00**	49.70	57.50	58.00
		AMI	51.50	55.50	55.50	38.60	**72.50**	52.40	67.90	55.50	63.20	64.60
	colour	ARI	23.00	0.10	0.10	49.00	55.90	79.20	87.40	88.50	**90.00**	88.00
		AMI	33.40	0.10	0.10	43.30	28.30	73.70	82.30	82.70	**84.50**	83.00
Cards	rank	ARI	30.10	29.70	26.70	35.70	33.10	24.30	24.70	33.00	**50.70**	34.70
		AMI	47.80	47.60	41.70	55.20	52.40	41.30	41.60	50.00	**68.40**	50.20
	suit	ARI	25.90	1.10	3.80	10.60	14.30	19.70	29.60	25.90	28.30	**29.70**
		AMI	34.40	1.20	8.20	16.60	19.80	27.40	37.10	33.60	35.40	**38.30**
NR-Objects	shape	ARI	95.30	94.40	94.40	76.00	72.70	65.70	94.50	75.90	95.00	**100.00**
		AMI	96.20	95.10	95.10	82.20	82.70	71.30	96.50	79.40	95.80	**100.00**
	material	ARI	0.00	26.70	30.60	30.70	**31.60**	9.20	1.60	14.80	0.00	9.00
		AMI	0.00	25.90	38.80	32.70	**39.40**	10.10	1.80	15.00	0.00	17.10
	colour	ARI	9.70	87.00	75.10	50.40	45.70	66.80	91.10	81.20	83.70	**97.80**
		AMI	21.70	93.00	79.00	65.70	66.00	81.20	95.30	88.30	91.40	**97.90**
Avg.		ARI	31.81	39.19	37.98	39.04	39.69	40.53	51.62	47.94	52.67	**54.50**
		AMI	41.51	45.98	45.89	48.93	50.30	48.10	57.80	54.04	58.68	**60.31**

Table 2: Main results table. Best results are in bold, second best results are underlinded.

4.1.4 Baselines

We use multiple image-based alternative clustering baselines and baselines using the generated text. It is important to note that the generated text uses additional information in the form of prompts provided by a user. While this implies that there is no exact comparison between image- and text-based methods, it is also worth noting that it is not possible to incorporate such information into image-based methods trivially. The code is implemented using the ClustPy[6] library (Leiber et al., 2023). Additional details are given in Appendix B.

Orth (Cui et al., 2007) iteratively identifies several clusterings by first clustering using PCA (keeping 90% of the variance) in combination with k-means and then creating a new orthogonal feature space. There are two strategies for orthogonalization: *orthogonal clustering* (orth-1) and *clustering in orthogonal subspaces* (orth-2).

Nr-Kmeans (Mautz et al., 2018) simultaneously optimizes several clusterings by assigning each clustering result a separate subspace in which k-means is executed.

ENRC (Miklautz et al., 2020) is a deep clustering method that assigns multiple clusterings to a dataset by (soft-)assigning each dimension of the embeddings space to a clustering.

Avg. Prompt. We measure the performance of clustering each text generated per prompt and subsequentially report the average performance.

Concat. by Category. We manually group all prompts together that belong to the same clustering type (e.g., "rank" or "suit"), concatenate all generated text, and cluster it using k-means.

[6] https://github.com/collinleiber/ClustPy

			TF-IDF				SBERT			
			concatenation		consensus		concatenation		consensus	
			min	max	min	max	min	max	min	max
Fruits-360	fruit	ARI	20.80	**24.60**	17.40	19.50	19.60	17.20	15.80	18.60
		AMI	36.60	**39.10**	29.00	32.90	33.00	30.90	23.20	26.90
	colour	ARI	51.80	52.20	51.10	51.90	**58.60**	57.20	54.30	54.70
		AMI	61.70	62.20	60.70	61.40	**71.50**	66.40	64.90	64.80
GTSRB	type	ARI	52.70	**73.20**	49.70	54.10	50.80	58.00	51.60	58.00
		AMI	60.80	**75.40**	56.60	60.40	57.80	64.10	60.00	64.60
	colour	ARI	74.70	74.00	87.10	87.30	73.10	70.20	**88.90**	88.00
		AMI	70.70	70.10	81.60	81.80	70.20	68.60	**83.20**	83.00
Cards	rank	ARI	28.60	28.00	27.00	26.90	40.30	**56.00**	36.30	34.70
		AMI	48.30	47.00	47.60	46.20	58.50	**72.20**	51.30	50.20
	suit	ARI	19.40	20.10	21.60	21.10	19.60	19.90	22.40	**29.70**
		AMI	29.30	28.90	23.60	23.70	30.50	27.30	27.40	**38.30**
NR-Objects	shape	ARI	98.70	98.90	99.30	99.30	**100.00**	**100.00**	**100.00**	**100.00**
		AMI	97.60	97.90	98.90	98.70	**100.00**	**100.00**	**100.00**	**100.00**
	material	ARI	**23.10**	22.40	0.10	1.00	0.00	0.00	9.00	9.00
		AMI	**22.70**	22.20	0.10	1.80	0.00	0.00	17.10	17.10
	colour	ARI	33.30	33.30	80.00	84.10	43.60	43.60	**97.80**	**97.80**
		AMI	65.20	65.20	87.50	90.10	66.60	66.60	**97.90**	**97.90**
Avg.		ARI	44.79	47.41	48.14	49.47	45.07	46.90	52.90	**54.50**
		AMI	54.77	56.44	53.96	55.22	54.23	55.12	58.33	**60.31**

Table 3: An ablation analysis of TGAICC, where "min" and "max" refer to the thresholding strategy, and concatenation and consensus to the aggregation scheme. Consensus-max resembles TGAICC. The best results are in bold, and the second best results are underlined.

4.2 Main Experiments

The results of our main experiments are shown in Table 2. They reveal that, on average, text-based methods, including TGAICC, outperform image-based methods. Further, we observe that TGAICC, on average, demonstrates superiority over average prompting and concatenation baselines. In addition, we can see that clustering by material in the NR-Objects dataset does not work in the text domain. See Table 6 for VQA examples. The main take-

	Prompt	TF-IDF ARI	TF-IDF AMI	SBERT ARI	SBERT AMI
suit	Can you tell me the suit of the playing card shown in the picture?	25.42	31.25	25.42	31.25
	What suit does the playing card in the image belong to?	25.59	33.53	25.59	33.53
	Could you identify the suit of the playing card depicted in the photo?	**29.29**	33.64	**29.29**	33.64
	Can you tell me the suit of the playing card shown in the picture? Answer concisely.	24.25	**37.32**	24.25	**37.32**
	What suit does the playing card in the image belong to? Answer concisely.	28.97	36.35	28.97	36.35
	Could you identify the suit of the playing card depicted in the photo? Answer concisely.	21.85	29.71	21.85	29.71
rank	Can you tell me the rank of the card shown in the picture?	26.76	43.33	26.76	43.33
	What is the numerical or face value of the card displayed in the image?	32.06	47.36	32.06	47.36
	What level or position does the card in the photo hold?	31.52	47.28	31.52	47.28
	Can you tell me the rank of the card shown in the picture? Answer concisely.	37.48	55.42	37.48	55.42
	What is the numerical or face value of the card displayed in the image? Answer concisely.	**38.79**	**56.09**	**38.79**	**56.09**
	What level or position does the card in the photo hold? Answer concisely.	31.15	50.44	31.15	50.44

Table 4: Ablation study comparing the clustering performance of individual prompts. Here we show a case study based on the Cards dataset. The best results are in bold, and the second-best results are underlined.

away is that, in many cases, the VQA model provides too much information, even information that should be used for a different clustering, e.g., color or shape. This highlights a core limitation of our methodology. If the text generation does not work sufficiently well, the subsequent clustering can not work. Nevertheless, TGAICC is model-agnostic and can be used with any VQA image-to-text system. In this way, it can use future advancements in VQA models.

4.3 Aggregation ablation

In this ablation study, we investigate the aggregation components of TGAICC. More specifically, we investigate the impact of the thresholding and aggregation strategies on clustering performance.

Setup. We ablate the "min" and "max" thresholding strategies, which find the minimum and maximum threshold such that the number of clustering groups corresponds to the expected number of alternative clusterings. We experiment with the consensus-clustering-based aggregation scheme used in TGAICC and compare it to the simple "concatenation" baseline, which concatenates the text of the corresponding clustering groups. Results are shown in Table 3. Note that TGAICC is consensus-max.

Results. Our analysis reveals that consensus clustering outperforms concatenation-based selection. Furthermore, SBERT-based clustering outperforms TF-IDF-based clustering. We observe that the performance of the 'min' and the 'max' strategies are very similar, indicating the stability of the method w.r.t. the thresholding strategy.

Suit		Rank	
Truth	Top Words	Truth	Top Words
heart	heart	ace	ace
diamond	diamond	king	king
club	club	queen	queen
spade	spade	jack	jack
		5	heart
		9	spade
		3	rank
		4	club
		6	diamond
		10	10
		2	twos
		8	8
		7	7

Table 5: This table shows how we are able to explain the datasets by listing the top most used words of the two final clusterings. For each top word, we show the ideal ground truth cluster name assignment.

4.4 Individual prompt analysis

TGAICC is based on the aggregation of multiple clusterings, which in turn are based on generated texts using different VQA prompts. As known from other tasks (Sclar et al., 2024), different prompts potentially result in high-performance variance.

Setup. In Table 4 we analyze the clustering performance per prompt on the case study of the Cards dataset. Note that again, we execute k-means 10 times and present the average results.

Results. In the case study, the addition of the prompt "Answer concisely" mostly yields similar clustering results to the original version, with a slight performance advantage when the "Answer

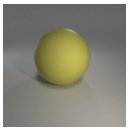
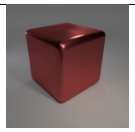
Material	Image	Question	Generated Text
Rubber		What substance is the item in the picture made of? Answer concisely.	Plastic
		What substance is the item in the picture made of?	The item in the picture appears to be a sphere made of a smooth, shiny material that could be plastic, metal, or a similar synthetic material. It's difficult to determine the exact material without more information or a closer inspection.
Metal		What substance is the item in the picture made of? Answer concisely.	The item in the picture is made of metal.
		Can you identify the material used in the object shown in the image? Answer concisely.	The object in the image appears to be made of a shiny, reflective material, possibly metal or a metal-like material.
		What is the composition of the object depicted in the photo? Answer concisely.	The object in the photo is a 3D rendering of a red cube.

Table 6: Some VQA examples on the NR-Objects dataset. While all answers are reasonable, they often provide too much information, such as information about the shape, or make statements about the ambiguity of the underlying material.

concisely" prompt is included. Furthermore, we note a significant variation in clustering performance across different prompts.

4.5 Explainability

Our cluster explainability aims to provide insights into the different clustering possibilities. This understanding is essential for interpreting and validating clustering outcomes. We use a simple word frequency-based algorithm to explain the data.

Setup. For a specific final clustering of TGAICC, we concatenate all generated texts belonging to the prompts used to obtain this clustering. The resulting text is changed to lowercase and made singular. Finally, to explain a final clustering, we determine the z most frequently occurring words, where z is the number of clusters of the respective clustering. For instance, for the suit clustering $z = 4$. Table 5 shows the resulting words for the Cards dataset. We reorder the ground truth cluster names suitably.

Analysis. Notably, the explainability method effectively identifies the "suits" cluster names, providing a comprehensive description of this clustering type, even though clustering performance has an AMI of less than 40%. Additionally, the frequency analysis exposed many of the card types in the dataset. However, suit names are also assigned as cluster names for the expected card ranking clusters (e.g., "heart" as the top word of the "5" cluster). Figure 6 presents concrete examples demonstrating that VQA models often provide additional information, such as suit, thereby explaining the inclusion of suits as rank names.

5 Discussion

5.1 Text-driven data interaction

Textual data, as a fundamental form of human communication, offers a natural and intuitive interface for interacting with complex datasets. Our method capitalizes on this inherent connection by utilizing textual prompts for VQA models to guide alternative clusterings. This approach aligns with real-world scenarios where users possess domain knowledge and seek answers to specific questions. We envision a future where users can explore datasets from diverse perspectives and test emerging hypotheses interactively using text. This research contributes towards this vision.

5.2 Domain Expertise

Our approach incorporates domain expertise, recognizing that users often either have specific questions or some knowledge about their data. This stands in contrast to the traditional clustering setup, which typically operates without user input. By leveraging domain knowledge, our approach aligns with real-world scenarios and allows for more targeted and insightful data exploration.

6 Conclusion

In conclusion, this research introduces TGAICC (Text-Guided Alternative Image Consensus Clustering), a novel approach that leverages prompting to inject domain knowledge and human intuition into the clustering process. The experiments on four common alternative image clustering benchmarks demonstrate that TGAICC outperforms competitive image- and text-based baselines. Furthermore,

the inherent explainability of text enables a deeper understanding of the underlying data cluster formations.

By utilizing textual prompts, we can explicitly guide the clustering process from various angles simultaneously, aligning with human intuition. This approach offers a more comprehensive and flexible way to analyze visual data, revealing insights that might be missed by traditional clustering methods.

7 Acknowledgements

This research was funded by the WWTF through the project "Knowledge-infused Deep Learning for Natural Language Processing" (WWTF Vienna Research Group VRG19-008) and through the project "Transparent and Explainable Models" (WWTF ICT19-041).

References

Nikolas Adaloglou, Felix Michels, Hamza Kalisch, and Markus Kollmann. 2023. Exploring the limits of deep image clustering using pretrained models. In *34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023*. BMVA.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikoł aj Bińkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. 2022. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 23716–23736. Curran Associates, Inc.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesht Sharma, Taewoon Kim, M Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Xiangru Tang, Mike Tian-Jian Jiang, and Alexander M. Rush. 2022. Promptsource: An integrated development environment and repository for natural language prompts.

Ying Cui, Xiaoli Z. Fern, and Jennifer G. Dy. 2007. Non-redundant multi-view clustering via orthogonalization. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA*, pages 133–142. IEEE Computer Society.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255.

Absalom E. Ezugwu, Abiodun M. Ikotun, Olaide O. Oyelade, Laith Abualigah, Jeffery O. Agushaka, Christopher I. Eke, and Andronicus A. Akinyelu. 2022. A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. *Engineering Applications of Artificial Intelligence*, 110:104743.

Xiaoli Zhang Fern and Carla E. Brodley. 2004. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 36, New York, NY, USA. Association for Computing Machinery.

Xifeng Guo, Long Gao, Xinwang Liu, and Jianping Yin. 2017. Improved deep embedded clustering with local structure preservation. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1753–1759. ijcai.org.

Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. 2013. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. 2021. A survey on contrastive self-supervised learning. *Technologies*, 9(1).

Sehyun Kwon, Jaeseung Park, Minkyu Kim, Jaewoong Cho, Ernest K. Ryu, and Kangwook Lee. 2024. Image clustering conditioned on text criteria. In *The Twelfth International Conference on Learning Representations*.

Johannes Lehner, Benedikt Alkin, Andreas Fürst, Elisabeth Rumetshofer, Lukas Miklautz, and Sepp Hochreiter. 2023. Contrastive tuning: A little help to make masked autoencoders forget. *arXiv preprint arXiv:2304.10520*.

Collin Leiber, Lukas Miklautz, Claudia Plant, and Christian Böhm. 2023. Benchmarking deep clustering algorithms with clustpy. In *2023 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 625–632. IEEE.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. 2023a. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. 2022. BLIP: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 12888–12900. PMLR.

Tao Li and Chris Ding. 2008. *Weighted Consensus Clustering*, pages 798–809.

Yunfan Li, Peng Hu, Zitao Liu, Dezhong Peng, Joey Tianyi Zhou, and Xi Peng. 2021. Contrastive clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8547–8555.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023b. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023a. Improved baselines with visual instruction tuning.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. In *NeurIPS*.

Stuart P. Lloyd. 1982. Least squares quantization in PCM. *IEEE Trans. Inf. Theory*, 28(2):129–136.

Dominik Mautz, Wei Ye, Claudia Plant, and Christian Böhm. 2018. Discovering non-redundant k-means clusterings in optimal subspaces. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1973–1982. ACM.

Lukas Miklautz, Dominik Mautz, Muzaffer Can Altinigneli, Christian Böhm, and Claudia Plant. 2020. Deep embedded non-redundant clustering. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04):5174–5181.

Lukas Miklautz, Martin Teuffenbach, Pascal Weber, Rona Perjuci, Walid Durani, Christian Böhm, and Claudia Plant. 2022. Deep clustering with consensus representations. In *2022 IEEE International Conference on Data Mining (ICDM)*, pages 1119–1124.

Emmanuel Müller, Stephan Günnemann, Ines Färber, and Thomas Seidl. 2012. Discovering multiple clustering solutions: Grouping objects in different views of the data. In *Proceedings of the 28th ICDE, Washington, DC, USA, 1-5 April, 2012*, pages 1207–1210.

Horea Mureșan and Mihai Oltean. 2018. Fruit recognition from images using deep learning. *Acta Universitatis Sapientiae, Informatica*, 10:26–42.

Daniel Müllner. 2011. Modern hierarchical, agglomerative clustering algorithms.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl,

Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR.

William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Karen Sparck Jones. 1988. *A statistical interpretation of term specificity and its application in retrieval*, page 132–142. Taylor Graham Publishing, GBR.

Andreas Stephan, Lukas Miklautz, Kevin Sidak, Jan Philip Wahle, Bela Gipp, Claudia Plant, and Benjamin Roth. 2024. Text-guided image clustering. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2960–2976, St. Julian's, Malta. Association for Computational Linguistics.

Alexander Strehl and Joydeep Ghosh. 2002. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617.

J.W. Tukey. 1970. *Exploratory Data Analysis*. Number v. 1 in Exploratory Data Analysis. Addison Wesley Publishing Company.

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research*, 11(95):2837–2854.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Junyuan Xie, Ross B. Girshick, and Ali Farhadi. 2016. Unsupervised deep embedding for clustering analysis. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 478–487. JMLR.org.

Jiawei Yao, Qi Qian, and Juhua Hu. 2024. Multi-modal proxy learning towards personalized visual multiple clustering. *arXiv preprint arXiv:2404.15655*.

Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. 2023. A survey on multimodal large language models. *arXiv preprint arXiv:2306.13549*.

Guoxian Yu, Liangrui Ren, Jun Wang, Carlotta Domeniconi, and Xiangliang Zhang. 2024. Multiple clusterings: Recent advances and perspectives. *Computer Science Review*, 52:100621.

A Datasets

In addition to the dataset description presented in Section 4.1, we provide the following supplemen-

tary materials to enhance the reader's understanding. Table 7 shows examples of images from each dataset. Furthermore, Table 7 provides all prompts generated by GPT-4, paired with their corresponding ground truth cluster names for each clustering type. Together, they give a good insight into the datasets and a textual interaction with them.

B Baselines

In the following, additional details for the baselines are given. We employ all of the following ones on the image embedding of the CLIP encoder of LLaVA-NeXT:

K-means. For all k-means runs, we utilize the k-means++ initialization strategy and set the number of initializations to 1. The code was implemented using scikit-learn (Pedregosa et al., 2011).

Nr-Kmeans. We set a limit of 300 maximum iterations.

Orth 1/2. We set the explained variance parameter to 90%.

ENRC. We try the learning rates $lr = 0.001, 0.0001$, use NR-Kmeans as initialization, a batch size of 128, and optimize for 200 epochs.

Dataset	Type	Cluster Names	Prompts
Fruits-360	fruit	apple, banana, cherry, grape	What kind of produce is shown in the picture? Can you identify the type of produce depicted in the image? What category of produce does the image represent? What kind of produce is shown in the picture? Answer concisely. Can you identify the type of produce depicted in the image? Answer concisely. What category of produce does the image represent? Answer concisely.
	colour	burgundy, green, red, yellow	Can you tell me the color of the fruits and vegetables shown in the picture? What color is the produce displayed in the photo? What hue are the items in the picture? Can you tell me the color of the fruits and vegetables shown in the picture? Answer concisely. What color is the produce displayed in the photo? Answer concisely. What hue are the items in the picture? Answer concisely.
GTSRB	type	70_limit, dont_overtake, go_right, go_straight	What kind of traffic sign is shown in the picture? Can you identify the category of the traffic sign displayed in the image? What class of traffic sign is depicted in the photo? What kind of traffic sign is shown in the picture? Answer concisely. Can you identify the category of the traffic sign displayed in the image? Answer concisely. What class of traffic sign is depicted in the photo? Answer concisely.
	colour	blue, red	What color is the traffic sign shown in the picture? Can you tell me the color of the traffic sign depicted in the image? What hue is the traffic sign in the photograph? What color is the traffic sign shown in the picture? Answer concisely. Can you tell me the color of the traffic sign depicted in the image? Answer concisely. What hue is the traffic sign in the photograph? Answer concisely.
NR-Objects	shape	cube, cylinder, sphere	Can you identify the form of the object shown in the picture? What form does the object in the picture take? Could you tell me the configuration of the object depicted in the image? Can you identify the form of the object shown in the picture? Answer concisely. What form does the object in the picture take? Answer concisely. Could you tell me the configuration of the object depicted in the image? Answer concisely.
	material	metal, rubber	What substance is the item in the picture made of? Can you identify the material used in the object shown in the image? What is the composition of the object depicted in the photo? What substance is the item in the picture made of? Answer concisely. Can you identify the material used in the object shown in the image? Answer concisely. What is the composition of the object depicted in the photo? Answer concisely.
	colour	blue, gray, green, purple, red, yellow	What color is the item shown in the picture? Can you tell me the color of the object depicted in the image? What hue does the object in the photo have? What color is the item shown in the picture? Answer concisely. Can you tell me the color of the object depicted in the image? Answer concisely. What hue does the object in the photo have? Answer concisely.
Cards	rank	ace, eight, five, four, jack, king, nine, queen, seven, six, ten, three, two	Can you tell me the rank of the card shown in the picture? What is the numerical or face value of the card displayed in the image? What level or position does the card in the photo hold? Can you tell me the rank of the card shown in the picture? Answer concisely. What is the numerical or face value of the card displayed in the image? Answer concisely. What level or position does the card in the photo hold? Answer concisely.
	suit	clubs, diamonds, hearts, spades	Can you tell me the suit of the playing card shown in the picture? What suit does the playing card in the image belong to? Could you identify the suit of the playing card depicted in the photo? Can you tell me the suit of the playing card shown in the picture? Answer concisely. What suit does the playing card in the image belong to? Answer concisely. Could you identify the suit of the playing card depicted in the photo? Answer concisely.

Table 7: Overview of the datasets, the names of their ground truth clusterings, and all generated prompts.

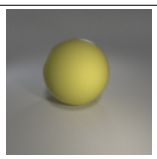
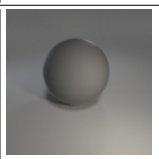
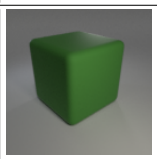
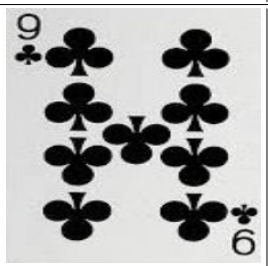
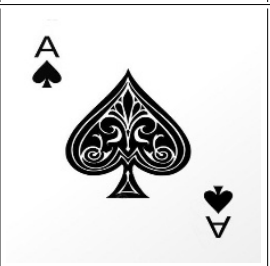
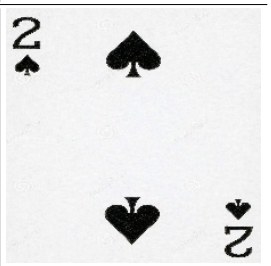
dataset	Image 1	Image 2	Image 3
Fruits-360			
GTSRB			
NR-Objects			
Cards			

Table 8: A few example images for each dataset.

9. From Calculation to Adjudication: Examining LLM Judges on Mathematical Reasoning Tasks

Title	From Calculation to Adjudication: Examining LLM judges on Mathematical Reasoning Tasks
Authors	Andreas Stephan, Dawei Zhu, Matthias Aßenmacher, Xiaoyu Shen and Benjamin Roth
Publication Outlet	In Review
Thesis-Reference	Stephan et al. (2024c)

Supplementary Material

The code is available on Github at `https://github.com/AndSt/llm_judges`.

Author Contributions

Andreas Stephan	conceived the initial idea, developed and expanded the initial idea. He designed the experiments, implemented the code, conducted the experiments, analyzed them, and iterated on them. Further, he wrote the majority of the paper.
Dawei Zhu	participated in developing and expanding the initial idea, provided feedback on the experimental design and experiments, and helped write the paper.
Matthias Assenmacher	participated in developing and expanding the initial idea, provided feedback on the experimental design and experiments, and helped write the paper.
Xiaoyu Shen	provided feedback on the experimental design and experiments, and, in general, provided mentoring and supervision.
Benjamin Roth	participated in developing and expanding the initial idea, provided feedback on the experimental design and the experiments, revised the draft of the paper, and, in general, provided mentoring and supervision.

From Calculation to Adjudication: Examining LLM Judges on Mathematical Reasoning Tasks

Andreas Stephan[1,2], Dawei Zhu[4], Matthias Aßenmacher[6,7], Xiaoyu Shen[5], Benjamin Roth[1,3]

[1]Faculty of Computer Science, [2]UniVie Doctoral School Computer Science,
[3]Faculty of Philological and Cultural Studies, University of Vienna, Vienna, Austria
[4]Saarland University, Saarland Informatics Campus, [5]Eastern Institute of Technology, Ningbo
[6]Department of Statistics, LMU Munich, [7]Munich Center for Machine Learning (MCML)

Correspondence: andreas.stephan@univie.ac.at

Abstract

To reduce the need for human annotations, large language models (LLMs) have been proposed as judges of the quality of other candidate models. The performance of LLM judges is typically evaluated by measuring the correlation with human judgments on generative tasks such as summarization or machine translation. In contrast, we study LLM judges on mathematical reasoning tasks. These tasks require multi-step reasoning, and the correctness of their solutions is verifiable, enabling a more objective evaluation. We perform a detailed performance analysis and find that easy samples are easy to judge, and difficult samples are difficult to judge. Our analysis uncovers a strong correlation between judgment performance and the candidate model task performance, indicating that judges tend to favor higher-quality models even if their answer is incorrect. As a consequence, we test whether we can predict the behavior of LLM judges using simple features such as part-of-speech tags and find that we can correctly predict 70%-75% of judgments. We conclude this study by analyzing practical use cases, showing that LLM judges consistently detect the on-average better model but largely fail if we use them to improve task performance. [1]

1 Introduction

The automatic evaluation of machine learning models promises to reduce the need for human annotations. Specifically, the LLM-as-a-judge paradigm (Zheng et al., 2023) has gained traction, aiming to assess or compare the quality of generated texts automatically. This approach is beneficial for automated data labeling (Tan et al., 2024), self-improvement of LLMs (Wu et al., 2024), and ranking the capabilities of LLMs, potentially concerning specific tasks (Zheng et al., 2023). Much like judges in the real world, who are expected to be exact, fair, and unbiased (Bangalore Principles, 2002),

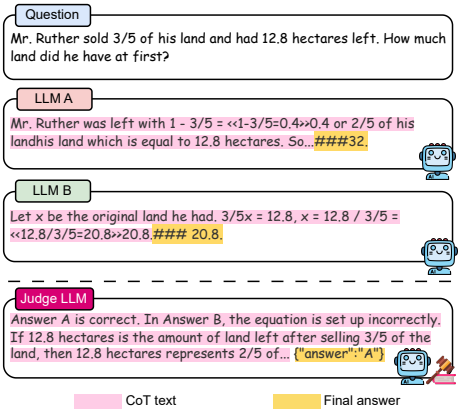


Figure 1: In our problem setup two LLMs (A and B), provide candidate answers for a math problem, and a judge LLM *has* to decide which one is correct. All three use chain-of-thought (CoT) reasoning (Wei et al., 2022).

LLM judges, should be unbiased and logical. Previous works investigate properties and biases of LLM judges on generative tasks such as translation or summarization (Kim et al., 2024b; Liu et al., 2024), typically evaluated using correlation with human annotators, and thus being inherently subjective.

In this work, we investigate LLM judges on mathematical reasoning datasets. Such tasks require complex multi-step reasoning and judgments can be analyzed through the lense of verifiable solutions, allowing us to investigate the relationship between judge and candidate models in a principled manner. In our setup, LLM Judges are given two answers and they have to classify whether both answers, one of them (which one), or none is correct (see Figure 1. We base our analysis on four large ($> 27B$ parameters) LLMs and four small ($< 10B$ parameters) LLMs on three mathematical reasoning datasets.

Our experiments contain a detailed performance examination. We find that the best tested judge is LLama 3.1 70B, reaching 60% to 90% judgment

[1]Code will be made available upon acceptance.

performance, depending on the dataset. Our results confirm the intuition that judgment performance is aligned with task difficulty.

We perform a statistical analysis of judgment performance and model quality. We find that the individual task performances of judge and candidate models are highly indicative features of judgment performance, as they explain most of the variance in a linear model (as measured by R^2). On the subset of questions where the candidate models give one correct and one incorrect answer, we uncover an intriguing correlation between judge performance and candidate models' task performance, indicating that LLM judges tend to select incorrect answers from better models.

We hypothesize that judges partially base their judgment on linguistic cues rather than solely on the reasoning withinin the answers. We follow literature analysing machine-generated text (Shaib et al., 2024) and find that 70%-75% of the judgments can be predicted using simple linguistic features, highlighting the systematicity behind the judge decisions.

Lastly, we analyze practical use cases and discuss usage recommendations. Our experiments suggest that LLM judges reliably detect the model of higher task performance but can not reliably improve task performance. Rather, we find that it is more sensible to use the judge model as an answer generator, and subsequently take the majority vote of all three answers.

In summary, our contributions are as follows:

1. We perform an in-depth performance analysis of LLM judges on three diverse mathematical reasoning tasks.

2. We identify a correlation between model quality, as measured by task performance, and judgment performance, indicating that LLM judges are biased towards higher-quality models.

3. We are able to predict the LLM judgment with 70%-75 accuracy using only stylistic patterns, e.g. N-grams of POS-tags. This indicates that LLMs, to a large degree, judge independently of the reasoning.

4. We find that judges reliably detect the model of higher quality but are not able to reliably improve task performance.

2 Related Work

2.1 LLM as Judges

Using *LLMs as judges* to evaluate text generated by LLMs, including their own outputs, has recently attracted significant interest because it reduces the need for human annotation (Zheng et al., 2023). Typically, large state-of-the-art models are used as judges. Applications include the automatic assessment of language model capabilities and, such as ranking models with respect to their competence on a given task (Zheng et al., 2023), and reinforcement learning from AI feedback by automatically generating data for preference optimization (Bai et al., 2022; Wu et al., 2024).

Various methods exist to make judgments (Zheng et al., 2023; Liusie et al., 2024). One approach is pairwise selection (Wang et al., 2024b), where two answers are presented, and the model is asked to select the better one. Another approach is pointwise grading (Li et al., 2024), where the model is asked to assign a grade based on a predefined scale, and the answer with a better grade is chosen. Judgment prompts may involve reference solutions or not. Another body of research explicitly trains models to act as judges (Kim et al., 2024a; Wang et al., 2024b) or closely related, as reward models (Wang et al., 2024c; Li et al., 2024).

The effectiveness of LLMs as judges is typically assessed by measuring the correlation or overlap with human judgments (Zheng et al., 2023; Kim et al., 2024b). In contrast, we focus on tasks with a concrete final answer. Finally, we want to stress that several works caution against the use of LLM judges as experts (Bavaresco et al., 2024; Koo et al., 2023; Raina et al., 2024; Doddapaneni et al., 2024).

2.2 Biases in LLM-as-a-judge

Human-annotated data inherently reflects the annotators' biases and opinions. These biases can be detrimental or (intentionally) beneficial, depending on the goals of the annotation process (Plank, 2022). Similarly, several studies have explored the biases present in LLM judges:

One linguistic bias is ordering bias (Zheng et al., 2023; Koo et al., 2023; Wang et al., 2024a), where a judge gives a different answer depending on the order in which answers are presented. Panickssery et al. (2024) note that it is possible to interpret position bias as a sign that the model is unsure. There are multiple works (Xu et al., 2024; Panickssery et al., 2024; Liu et al., 2024) that find evidence for

self-bias or self-preference. Koo et al. (2023) provide a benchmark for analyzing cognitive biases. West et al. (2024) and Oh et al. (2024) explore the "Generative AI Paradox" where it is easier for LLMs to generate solutions rather than analyzing them, unlike humans who often find analysis easier than generation.

In this work, we aim to establish a better understanding of underlying patterns that relate judgments to interpretable factors, such as task performance or stylistic patterns.

3 General Setup

In the following, we describe the problem setting, including the used notation, and the general experimental setting including used models and datasets.

3.1 Problem Description

In this work, we use an LLM judge, referred to as J, to assess answers produced by two other candidate LLMs, A and B, in response to math questions (see Figure 1 for an illustrative example). The two candidate answers may both be correct, both incorrect, or either the answer of model A or B correct. The judge's task is to determine which of these cases applies by reviewing both the CoT reasoning and the final responses provided in candidate answers.

Thus, the judge engages in a four-class classification task. We denote the judge's accuracy by the score $S^J_{A,B}$ and call this metric *judgment performance*. Further, we define the *task performance* of an individual model X on a specific dataset as S_X, e.g. S_A, S_B or S_J.

3.2 Datasets

The experiments encompass three mathematical reasoning datasets where models highly benefit from multi-step CoT reasoning. For all datasets, we use accuracy as the performance metric.

AQUA-RAT (Ling et al., 2017) is a dataset to test the quantitative reasoning ability of LLMs. Unlike the other two datasets, the questions are multiple-choice. **GSM8K** (Cobbe et al., 2021) consists of grade school math word problems. The answers are free-form numbers. **MATH** (Hendrycks et al., 2021) contains challenging competition mathematics problems. Find more details in Appendix A.1

3.3 Models

We evaluate the performance of openly available LLMs, including four large models including *Qwen 2.5 72B* (Yang et al., 2024), *Llama 3.1 70B* (AI@Meta, 2024), *Yi 1.5 34B* (Young et al., 2024), *Mixtral 8x7B* (Jiang et al., 2024) and four small models, namely *Llama 3 8B* (AI@Meta, 2024), *Gemma 1.1 7B* (Gemma Team et al., 2024), *Mistral 7B v0.3* (Jiang et al., 2023), and *Mistral 7B v0.1* (Jiang et al., 2023). We use the chat- or instruction-tuned model variants and test each model as a candidate answer generator and as a judge. More information is in Appendix A.2.

3.4 Text Generation

This section describes the generation of candidate answers and judgments. Find more information on prompts and hardware details in Appendix A.

Candidate answer generation. For each model we sample two CoT solutions using 4-shot prompting by setting the temperature to 0.9. By generating two answers a_1, a_2 from the same model, we can also evaluate judgments of two different answers by the same model.

Judgements. For all 36 unique model combinations (A, B)[2], each model as judge J and each sample of a dataset, we generate a zero-shot judgment. In the case of self-pairing, i.e., $A = B$, we use both generated candidate answers, a_1 and a_2. Otherwise, for consistency, we always use the same sampled candidate answer a_1. We accommodate positional bias (Zheng et al., 2023; Koo et al., 2023) by prompting in both possible orders. We obtain the judgment performance by averaging how often the judgments were correctly classified across orderings.

4 Performance Analysis

The experiments have multiple degrees of freedom, such as judges, candidate models, and datasets. To gain a comprehensive understanding of judges' behavior, we consider two perspectives. First, we investigate judge performance for each dataset, aiming to associate judge performance with task difficulty. Second, for a fixed dataset, we analyze judge performance across different pairs of candidate models.

4.1 General Performance

First, we compare how often the judges make a correct classification across different datasets and

[2]We consider all pairs from the eight LLMs, including self-pairing, yielding $\binom{8+2-1}{2} = 36$ combinations.

		Llama 3.1 70B	Qwen 2.5 72B	Qwen 2.5 14B	Gemma 2 27B	Qwen 2.5 7B	Gemma 2 9B	Llama 3.1 8B	Gemma 2 2B
(1) $S^J_{A,B}$	GSM8K	**90.05**	85.39	89.2	81.96	81.92	83.60	79.96	64.33
	AQUA-RAT	**74.47**	69.26	72.26	68.48	65.09	67.48	66.26	60.97
	MATH	61.18	58.03	**62.36**	55.34	50.70	52.92	50.96	50.35
(2) Same answer	GSM8K	95.27	**95.46**	95.02	92.27	92.86	94.70	88.13	75.50
	AQUA-RAT	**79.8**	77.74	77.19	78.67	77.21	77.94	74.52	76.01
	MATH	**79.09**	77.91	76.86	75.39	73.14	75.65	71.05	77.85
(3) Different Answer	GSM8K	**70.04**	55.67	68.43	47.09	49.93	49.50	51.41	31.71
	AQUA-RAT	57.44	48.92	**57.64**	45.55	40.45	46.31	44.10	30.76
	MATH	48.95	45.40	**51.47**	42.66	36.70	38.86	36.41	32.50
(4) 1-correct	GSM8K	**78.18**	64.08	76.92	52.25	56.19	58.10	59.26	27.78
	AQUA-RAT	66.43	57.10	**69.22**	44.44	47.13	54.43	52.93	24.05
	MATH	71.92	70.19	**79.62**	41.80	57.79	60.97	60.73	22.62

Table 1: Performance of judge LLMs (1) on all samples, (2) on samples where A and B agree, (3) on samples where A and B disagree and (4) on samples where exactly one given answer is correct. Results are averaged over all candidate model pairs (A, B). The highest accuracy is **bold** and the second highest underlined.

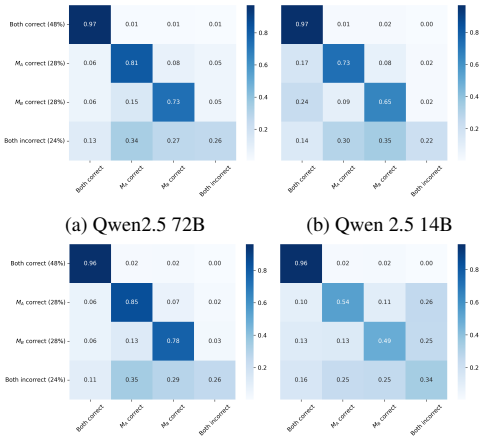
different subsets of the datasets.

Setup. We analyze multiple cases, each corresponding to a specific subset of the data. *Case (1)* investigates the observed judgment performance $S^J_{A,B}$ on the full dataset and *Case (2)* analyzes the subset where both models give the same answer $(A = B)$. *Case (3)* shows the performance where both models give a different answer $(A \neq B)$ and *Case (4)* describes the performance on the subset where exactly one answer is correct. The results are shown in Table 1. Further, we show the class confusion matrix for the four best-performing judges in Figure 2.

Results. In general, we observe in Table 1 that larger models outperform smaller models, with LLama 3.1 70B performing the best. Interestingly, Qwen 2.5 14B outperforms Qwen 2.5 72B. As shown in Figure 2, the LLM judges have a performance of larger than 95% if both answers are correct. Conversely, the most challenging situation is when both answers are incorrect. It seems that the difficulty of a problem also transcends the difficulty of making a judgment. This is not necessarily intuitive. For instance, humans may find it easier to detect individual wrong reasoning steps and identify wrong answers, respectively.

In cases where one answer is correct and one answer is incorrect, we observe a moderate performance of the judges, reaching up to 80% accuracy (see Case (4) in Table 1 and Figure 2).

In Case (3) where both answers disagree, we observe moderate performance for large models of up to 70%. Here, the smallest model Gemma 2 2B, has a low performance of around 35%. In what follows, we mostly focus on the analysis of the four largest LLMs as judges.



(a) Qwen2.5 72B (b) Qwen 2.5 14B

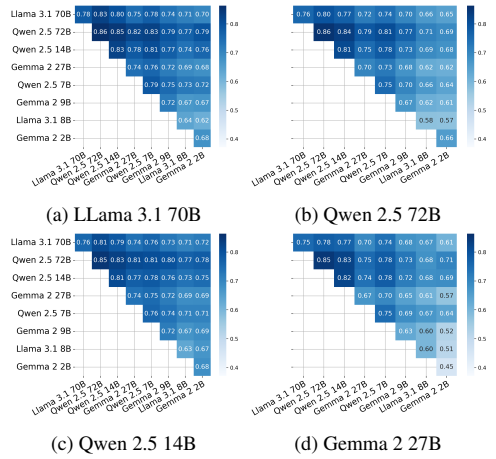
(c) Llama 3.1 70B (d) Gemma 2 27B

Figure 2: Class confusion matrices per model. We observe that it is difficult for judges to detect that both answers are incorrect.

4.2 Performance per model combination

Each model has unique strengths and weaknesses and often answers different questions correctly. In this section, we analyze the judgment performance per model pair to gain a better understanding of the impact of candidate model combinations on judgment performance.

Setup. Figure 3 illustrates the judgement performance $S^J_{A,B}$ across model pairs (A, B), indicating the probability of a correct judgement. The results are averaged over datasets and presented as an upper triangular matrix due to symmetry (we always present the answers in both possible orders and average performance). We report the performance of all models used as judges in the Appendix B in Table 9.



(a) LLama 3.1 70B (b) Qwen 2.5 72B

(c) Qwen 2.5 14B (d) Gemma 2 27B

Figure 3: Judgment Performance $S^J_{A,B}$ of LLM judges on model pairs, averaged across datasets.

Results. The highest performance is achieved when two answers of Qwen 2.5 72B are compared which is the highest performing model (see task performance in Appendix B.1) In general, we observe that it is easier for the judge to make a correct judgment if candidate models are of higher quality. This seems intuitive because such models likely structure and present their reasoning well, allowing a judge to compare solutions more easily. Figure 2 gives an additional explanation. It shows that judges very reliably detect whether both answers are correct. When both models are capable, it is more likely that both give a correct answer, which makes it easier for LLM judges to classify correctly.

Interestingly, the judgment performances of Qwen 2.5 72B, Qwen 2.5 14B, and LLama 3.1 70B are very similar across pairs. The former possibly agree on a lot because of the similarity of training data and knowledge distillation (Hinton et al., 2014). The largest performance difference is that LLama 3.1 70B performs 10% better when Qwen 2.5 72B and Gemma 2 2B are compared.

These results show that there is a relationship between the task performance of a candidate model and judgment performance. The following section will provide further analysis.

5 Population-level Analysis: Judgements and Model Quality

In this section, we investigate the relationship between LLM judgments and candidate LLM quality.

	Llama 3.1 70B	Qwen 2.5 72B	Qwen 2.5 14B	Gemma 2 27B
R^2	0.89	0.87	0.85	0.93
(p-value)	(0.0)	(0.0)	(0.0)	(0.0)

Table 2: R^2 values for the regression models *per judge* (first row) and corresponding p-values of the Overall F-Test (second row). All R^2 values are statistically significant on the 5% level.

First, we provide a statistical analysis where we use LLM task performance to explain the variance in LLM judgment performance. Further, we focus on the subsets where the candidate models make exactly one correct and one incorrect prediction. We observe a strong statistical relationship between the difference in candidate task performances and judgment performances.

5.1 Can we explain Judgement Performance using Task Performance?

A good indicator of the competence of a model on a specific dataset is its task performance. Clearly, there is a relationship between the quality of the involved models and the made judgments. We investigate the relationship between task performances (of candidate and judge models) and judgment performance.

Setup. We fit multiple different linear regression models using the judgment performances as the target variables Y, including all variations of judges, model pairs (A, B), and datasets D. Regarding the covariates $\mathbf{X}$ in the model, we solely use the task performances $S_X, X \in \{J, A, B\}$ of judge and candidate models, to predict judgment performance. Since we are not specifically interested in the individual features' effects, but rather in their ability to explain the variation of judgment performance, we rely on the coefficient of determination, R^2, for evaluation (Fahrmeir et al., 2013, see Appendix D).

Results. The results are shown in Table 2 (excluding data sets from the probability formulas for simplicity). We observe that the performance-related features of the models can explain the variation in the judgment performance (all R^2 values greater or equal than 0.85), very well. Logically, S_A and S_B, have significant[3] explanatory power for judgment performance, as they encompass all correct answers.

[3]We test statistical significance using an Overall-F-Test for each fitted model. Further details are in Appendix D.

5.2 Are LLM judges biased towards LLMs of higher quality?

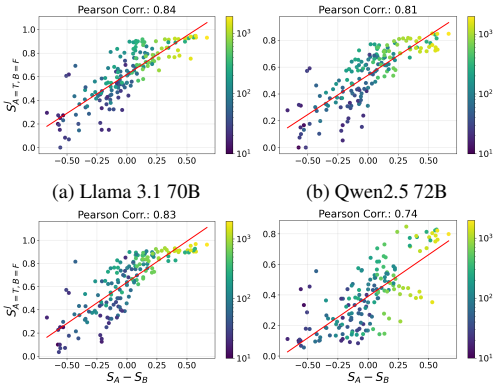
To get a better understanding of whether there is a bias of LLM judges towards LLMs of higher quality, we investigate the subset where one candidate answer is correct and the other candidate answer is incorrect. This subset is of the highest practical relevance. The goal is to investigate the relationship between the task performances of the candidate models and the judge's performance.

Setup. For all model pairs $(A, B), A \neq B$ we analyze subsets where A's solutions are correct, and B's solutions are incorrect, and call it 1-correct. Note that we can always order A and B this way. Each plot in Figure 4 shows the relationship between judge performance on the 1-correct subset (Y-axis) and *candidate model performance gap* of A and B, i.e., $S_A - S_B$ (X-axis). The color of the points indicate the size of the particular subset of samples. Examples of these subsets and their corresponding performances are in Appendix C.1.

Results. The analysis reveals a strong correlation (Pearson's $r^2 > 0.78$) between judgement performance and candidate model performance gap. For the rest of this section, we call the model of higher performance on a dataset the *more competent model*. I.e., if the performance gap is larger than 0, the model giving the correct answer (A) is the more competent model. If the correct model is the more competent model, the judgment performance on the subset is higher, e.g., for LLama 3.1 70B, sometimes approaching 100%. If the performance gap is more positive, it is easier to choose the correct answer. On the other hand, if the less competent model gives the correct answer, judgment performance is low, often lower than 20%.

We infer that LLM judges are biased towards models of higher task performance. This finding aligns with previous research identifying self-bias (Xu et al., 2024; Panickssery et al., 2024; Liu et al., 2024), as judge LLMs are typically of higher quality than the judged models. We hypothesize that this bias arises because more competent models articulate their responses more convincingly and exhibit a specific writing style, thereby misleading the judges.

However, models of higher task performance typically answer correctly more often (as indicated by the color of the points in Figure 4.



(a) Llama 3.1 70B (b) Qwen2.5 72B

(c) Qwen 2.5 14B (d) Gemma 2 27B

Figure 4: Judges' accuracy vs. performance gap between two candidate models A and B. Each point represents a subset where A is correct, and B is incorrect. The color reflects the size of these subsets.

6 Sample-level analysis: Judgments and Stylistic Patterns

In Section 5, we found that the quality of a candidate LLM (as indicated by the task performance) correlates with the made judgment. We hypothesize that models of higher quality exhibit a particular style of expressing themselves and judges partially base their judgment on the incorporated textual cues. Motivated by recent work in machine-generated text detection which finds that LLMs often exhibit certain styles (Wu and Aji, 2025) or patterns (Shaib et al., 2024), we aim to gain a deeper understanding of whether shallow or even content-independent patterns affect the final judgment.

Setup. We separate all judgments each judge made into training and test splits and train two classifiers. The test accuracy is reported in Table 3. We use two types of features. First, we use TF-IDF embeddings. Secondly, we use N-Grams of part-of-speech (POS) tags, motivated by Shaib et al. (2024) who show and investigate the distinct occurrence of such in LLM-generated text. Given two candidate answers, we create two independent feature sets and concatenate those. Then a logistic regression and a RandomForest classifier (Breiman, 2001) are trained on these concatenated features. Find more information in Appendix E.

Results. We observe that the models achieve a performance between approximately 70% and 75%. This indicates that structural information (POS

Features	Model	Llama 3.1 70B	Qwen 2.5 72B	Qwen 2.5 14B	Gemma 2 27B
POS	LR	72.79	69.66	72.33	70.19
	RF	71.71	69.77	71.89	69.18
TF-IDF	LR	75.75	73.65	75.12	72.27
	RF	75.65	71.05	75.79	70.58

Table 3: Accuracy of predicting LLM judges' decisions using Logistic Regression (LR) and Random Forest (RF) classifiers based on N-Grams of either POS tags or TF-IDF features.

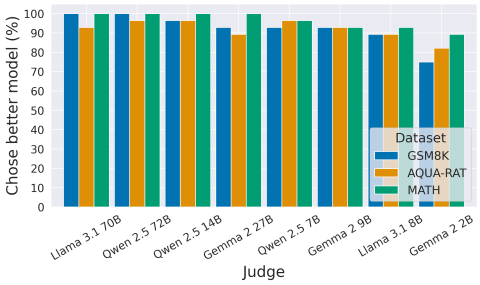


Figure 5: Percentage of model pairs (A, B) where a judge picks a better model A (meaning $S_A > S_B$), by selecting more answers of A than from B.

tags) and word choice (TF-IDF) are important factors in understanding the patterns behind the behavior of LLM judges. The ground truth judgment distribution is shown in Appendix E.

Nevertheless, these results suggest that decision-making is a multi-faceted process. While specific shallow cues hold influence, a substantial portion of the decision-making process (25%-30%) can not be predicted this way and is based on other contextual factors which could include reasoning or noise.

7 Usage recommendations

Lastly, we aim to give some usage recommendations. We start by analyzing two applied questions, namely, whether LLM judges can identify models of higher task performance and whether LLMs should be used to improve task performance. In the end, we discuss those results, connecting them to the overall insights of this paper.

7.1 Do judges identify better models?

An essential application of LLM judges is whether they can accurately identify which model performs better for a given task. This is crucial if we want to rank LLMs by their capabilities or if a practitioner wants to decide which model to deploy.

Setup. We evaluate which model a judge perceives as better by measuring the frequency of how often a judge selects the answer of a specific model. Formally, let (A, B) be a candidate model pair where we assume that A has higher task performance, i.e. $S_A > S_B$. If the judge chooses A more often, we say a judge correctly determines A to be better than B. For this analysis, we determine the proportion of model pairs (A, B) for which the judge chooses A over B for all pairs $(A, B), S_A > S_B$ as shown in Figure 5.

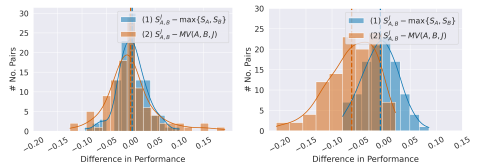
Results. We observe that all tested large models consistently select the more competent model, i.e., the model with higher task performance. Also, the small models with 7-9B parameters choose the correct model in over 90% of the cases. In general, it seems to be the hardest on the AQUA-RAT dataset. This is also the hardest dataset in Case (4), in Table 1, where exactly one answer is correct. Note that the bias found in Section 5.2 is not necessarily problematic for this specific use case, because a bias towards the more competent model supports a correct outcome of this experiment.

7.2 Do judges elicit task improvement?

Another interesting question of practical relevance is whether it makes sense to use LLM judges to improve task performance. One use case is the application of LLM judges in agentic systems where LLM judges might serve as a dedicated unit in a system. Another use case is the subsequent usage of the answers chosen by the judge for self-training (Yuan et al., 2024).

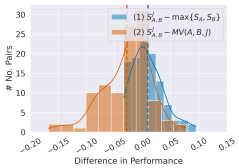
Setup. We separate the analysis into two questions. In Case (1), we evaluate whether the answers chosen by the judge result in a better performance than the individual models. Formally, for all pairs of models (A, B), we plot the difference of performance of chosen answers, $C^J_{A,B}$ and maximal single candidate model performance $\max\{S_A, S_B\}$ in blue in a bar chart in Figure 6. Secondly, in Case (2), we test whether it makes more sense to use the judge model to generate a candidate answer J and then take the majority vote across all three answers. Therefore we plot the performance difference of $C^J_{A,B} - \text{MV}(A, B, J)$ in orange in a bar chart, where $\text{MV}(A, B, J)$ is the performance of the majority vote across all three answers.

Results. In general, we observe that the performance differences are almost following a normal

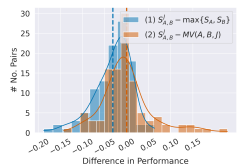


(a) LLama3.1 70B Full

(b) Qwen 2.5 72B Full



(c) Qwen 2.5 14B Full



(d) Gemma 2 27B

Figure 6: The Y-axis describes the number of model pairs A, B where the answers chosen by the judge achieve a higher task performance than the performances of the individual models (blue) or than the majority vote (MV) of answers of A, B and J as candidate answer generator (red). The X-axis describes the performance difference. A value of, e.g., $x = 0.05$ means the answers chosen by the judge result in a 5% (absolute) performance increase.

distribution. In Case (1), the distribution has a mean value (dashed line) slightly larger than 0 for LLama 3.1 70B (0.3) and Qwen 2.5 14B (0.9). That means that, on average, the answers chosen by the judge result in slightly increased performance, e.g., an increase from 40% accuracy to 40.9% accuracy. In Case (2), the mean value is never larger than 0, meaning that the majority vote is more likely to be better than the answer chosen by the judge. Especially for Qwen 2.5 14B and Qwen 2 72B, it is more viable to use the majority voting strategy.

7.3 Discussion

Our analysis of LLM judges on mathematical reasoning tasks reveals several insights for practitioners, which we discuss in the following. We separate our discussion into the sample level, i.e., the interpretation of a single prediction, and aggregate level, i.e., the interpretation of a set of predictions.

Sample level. In Table 1, we find that LLM judges often achieve a strong judgment performance ($S^J_{A,B} > 80\%$ accuracy) across tasks. While this is a solid classification performance, it means that the prediction is wrong in 20% of the cases which limits practical applicability. In Section 4, we observe that LLM judges demonstrate high precision when identifying correct answers from both models. This might be valuable for filtering samples and curating training data or, e.g., self-training. Nevertheless, one has to be careful how to use these because correctly judged samples are biased towards simple samples. In summary, we do not recommend fully relying on individual LLM judgments, especially not in high-stakes domains such as legal or health care.

Aggregate level. As shown in Figure 5, we find that LLM judges are consistently able to select or rank models by their task performance. This is supported by Section 5.1 where we show that a simple linear model can explain a high share of the variance in judgment performance, given individual task performances, suggesting that the performance difference of two candidate models is linearly linked to the judgment outcome.

In summary, our results suggest that LLM judges are more effective and consistent at aggregate-level comparisons than instance-level judgments, for example when ranking or selecting which LLM is better for a particular task when no ground truth data is available.

8 Conclusion

We conduct a thorough analysis of LLM judges on mathematical reasoning tasks. We evaluate the judgment performance of eight models of different sizes on three datasets. We find that larger judge models generally outperform smaller judge models and that judges can reliably detect whether both answers are correct. Our analysis reveals a strong correlation between judgments and task performance, indicating that judges tend to choose models of higher quality even if their answers are incorrect. We hypothesize that LLM judges partially base their decisions on linguistic cues in contrast to the reasoning within the answers. We support this hypothesis with our experiments showing that 70% of the judges' decisions can be predicted using simple linguistic features such as N-grams of part-of-speech tags. Lastly, our analysis finds that LLM judges reliably detect LLMs of higher task performance but are not reliably useable to improve task performance. Our results show that LLM judges contain biases and suggest that practitioners should not blindly trust LLM judges. We advise practitioners to carefully decide whether LLM judges should be used in their particular application.

With this work, we set the stage for further research to investigate how to understand, use, and improve LLM judges.

9 Limitations

Our analysis is primarily focused on mathematical reasoning datasets, which allows us to explore judgments through the lens of verifiability, i.e., problems that have a definitely correct answer. While this approach provides valuable insights, it limits the generalizability of our findings to other tasks or domains. Nevertheless, we want to emphasize the importance of the class of verifiable tasks. For instance, there is currently a focus on training so-called large reasoning models, which demonstrate significant progress in solving complex problems such as coding or maths. It is a possibility that an increased capability of LLMs on verifiable tasks fuels scientific progress.

In our experiments, we focus on testing a single, specific prompt. It is common knowledge that LLMs are highly sensitive to variations in prompt phrasing, which can substantially influence their performance. However, the resources available to us do not allow us to meet the computational demands necessary to run our experiments with multiple prompts. Further, our impression is that it is a custom approach to conduct LLM studies using single prompts, as they are typically indicative of behavior. Therefore we decided to run our analysis on full datasets with a single prompt instead of using subsets of datasets with variations of the prompt with mostly the same content.

References

AI@Meta. 2024. Llama 3 model card.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Bangalore Principles, 2002. 2002. The bangalore principles of judicial conduct. Available from the Judicial Integrity Group website.

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, André F. T. Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K Surikuchi, Ece Takmaz, and Alberto Testoni. 2024. Llms instead of human judges? a large scale empirical study across 20 nlp evaluation tasks.

Leo Breiman. 2001. Random forests. *Mach. Learn.*, 45(1):5–32.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Sumanth Doddapaneni, Mohammed Safi Ur Rahman Khan, Sshubam Verma, and Mitesh M Khapra. 2024. Finding blind spots in evaluator LLMs with interpretable checklists. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 16279–16309, Miami, Florida, USA. Association for Computational Linguistics.

Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, Brian Marx, Ludwig Fahrmeir, Thomas Kneib, Stefan Lang, and Brian Marx. 2013. *Regression models*. Springer.

Google Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open models based on gemini research and technology.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *NeurIPS*.

Geoffrey Hinton, Jeff Dean, and Oriol Vinyals. 2014. Distilling the knowledge in a neural network. In *NIPS 2014 Deep Learning Workshop*, pages 1–9.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024a. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, Dongkeun Yoon, Guijin Son, Yejin Cho, Sheikh Shafayat, Jinheon Baek, Sue Hyun Park, Hyeonbin Hwang, Jinkyung Jo, Hyowon Cho, Haebin Shin, Seongyun Lee, Hanseok Oh, Noah Lee, Namgyu Ho, Se June Joo, Miyoung Ko, Yoonjoo Lee, Hyungjoo Chae, Jamin Shin, Joel Jang, Seonghyeon Ye, Bill Yuchen Lin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024b. The biggen bench: A principled benchmark for fine-grained evaluation of language models with language models.

Ryan Koo, Minhwa Lee, Vipul Raheja, Jong Inn Park, Zae Myung Kim, and Dongyeop Kang. 2023. Benchmarking cognitive biases in large language models as evaluators.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Junlong Li, Shichao Sun, Weizhe Yuan, Run-Ze Fan, hai zhao, and Pengfei Liu. 2024. Generative judge for evaluating alignment. In *The Twelfth International Conference on Learning Representations*.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. 2017. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada. Association for Computational Linguistics.

Yiqi Liu, Nafise Sadat Moosavi, and Chenghua Lin. 2024. Llms as narcissistic evaluators: When ego inflates evaluation scores.

Adian Liusie, Potsawee Manakul, and Mark Gales. 2024. LLM comparative assessment: Zero-shot NLG evaluation through pairwise comparisons using large language models. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 139–151, St. Julian's, Malta. Association for Computational Linguistics.

Juhyun Oh, Eunsu Kim, Inha Cha, and Alice Oh. 2024. The generative AI paradox in evaluation: "what it can solve, it may not evaluate". In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 248–257, St. Julian's, Malta. Association for Computational Linguistics.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Barbara Plank. 2022. The "problem" of human label variation: On ground truth in data, modeling and evaluation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10671–10682, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Vyas Raina, Adian Liusie, and Mark Gales. 2024. Is llm-as-a-judge robust? investigating universal adversarial attacks on zero-shot llm assessment.

Skipper Seabold and Josef Perktold. 2010. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*.

Chantal Shaib, Yanai Elazar, Junyi Jessy Li, and Byron C Wallace. 2024. Detection and measurement of syntactic templates in generated text. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6416–6431, Miami, Florida, USA. Association for Computational Linguistics.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation: A survey.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. 2024a. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9440–9450, Bangkok, Thailand. Association for Computational Linguistics.

Yidong Wang, Zhuohao Yu, Wenjin Yao, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. 2024b. PandaLM: An automatic evaluation benchmark for LLM instruction tuning optimization. In *The Twelfth International Conference on Learning Representations*.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. 2024c. Helpsteer2: Open-source dataset for training top-performing reward models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, Benjamin Newman, Pang Wei Koh, Allyson Ettinger, and Yejin Choi. 2024. The generative AI paradox: "what it can create, it may not understand". In *The Twelfth International Conference on Learning Representations*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Minghao Wu and Alham Fikri Aji. 2025. Style over substance: Evaluation biases for large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 297–312, Abu Dhabi, UAE. Association for Computational Linguistics.

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston, and Sainbayar Sukhbaatar. 2024. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge.

Wenda Xu, Guanglei Zhu, Xuandong Zhao, Liangming Pan, Lei Li, and William Yang Wang. 2024. Pride and prejudice: Llm amplifies self-bias in self-refinement.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zhihao Fan. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Alex Young, Bei Chen, Chao Li, Chengen Huang, Ge Zhang, Guanwei Zhang, Heng Li, Jiangcheng Zhu, Jianqun Chen, Jing Chang, Kaidong Yu, Peng Liu, Qiang Liu, Shawn Yue, Senbin Yang, Shiming Yang, Tao Yu, Wen Xie, Wenhao Huang, Xiaohui Hu, Xiaoyi Ren, Xinyao Niu, Pengcheng Nie, Yuchi Xu, Yudong Liu, Yue Wang, Yuxuan Cai, Zhenyu Gu, Zhiyuan Liu, and Zonghong Dai. 2024. Yi: Open foundation models by 01.ai.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

A Experimental Setup

We provide further details on the general setup described in Section 3. Specifically, we include statistics and examples of the datasets, additional information on the models used, and the exact prompts employed in this study.

A.1 Datasets

Additional information about the datasets is given in Table 4, which presents an overview of the dataset statistics. Note that for the MATH dataset, we only include the most challenging questions, called levels 4 and 5, in the dataset. Notably, it has ground truth answer sequences that are, on average, almost three times longer than those in other datasets.

	# questions	Avg. # question characters	Avg. # answer characters
AQUA-RAT	254	239.1	203.1
MATH	1516	216.5	643.9
GSM8K	1319	239.9	292.9

Table 4: An overview of dataset size and text length.

User

You are a reasoning assistant. Always answer exactly in the same format. Use '####' to separate the final answer (without additional comments) from the reasoning.

« Few-Shot Question 1 »

Assistant

« Few-Shot Answer 1 »

...

...

User

« Few-Shot Question 4 »

Assistant

« Few-Shot Answer 4 »

User

« Sample Question »

Figure 7: The prompt to solve tasks. Few-shots and actual questions are filled in within "«" and "»" symbols.

In Table 5, we provide examples of questions and their corresponding answers from the ground truth. Note that these examples were used for few-shot prompting.

A.2 Models

We execute all models using the VLLM software for LLM serving (Kwon et al., 2023). The weights for all models are accessible through Huggingface Transformers (Wolf et al., 2020). Table 6 includes hyperlinks to each model for easy reference.

A.3 Prompts

We used two different prompts within this project. In general, we designed the prompts to be minimial, by assigning a minimal personality, a quick task description, and description of the output format.

User

Question:
« question »

Answer A:
« answer A »

Answer B:
« answer B »

Compare both answers in detail and decide whether both answers are correct, both answers are incorrect or whether answer 1 or answer 2 is correct.

Conclude with a JSON in Markdown format indicating your choice between "answer_1", "answer_2", "both_correct" or "both_incorrect":
```json
{
"answer": "..."
}
```

Figure 8: Judge Prompt. Candidate answers are filled in within "«" and "»" symbols.

The prompt shown in Figure 7 is used for the candidate solution generation for all datasets. Examples of the few-shots are in Table 5. The prompt for the judges is given in Figure 8. Note that we run experiments for both orders of the answers of the models A and B.

A.4 Infrastructure

The experiments were run on NVIDIA A100 and NVIDIA H100. The judgments used in Section 4 took around 3 day equivalents on 4 A100 40GB. Using 2 H100 90GB and 4 A100 40 GB it took less than 2 days.

B General Performance

This section provides additional information related to Section 4. Specifically, we present the task performance of all models across all datasets, as well as the judging performance of all models when used as judges.

B.1 Task Performance

In various contexts in this work, the task performance of the individual models is essential. Therefore, we provide the accuracy of all models and all datasets in Table 7.

	Question	Answer
AQUA-RAT	Two friends plan to walk along a 43-km trail, starting at opposite ends of the trail at the same time. If Friend P's rate is 15% faster than Friend Q's, how many kilometers will Friend P have walked when they pass each other? Options: A)21 B)21.5 C)22 D)22.5 E)23	If Q complete x kilometers, then P completes 1.15x kilometers. x + 1.15x = 43 2.15x=43 x = 43/2.15 = 20 Then P will have have walked 1.15*20=23 km. The answer is E. #### E
GSM8K	Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?	Natalia sold 48/2 = «48/2=24»24 clips in May. Natalia sold 48+24 = «48+24=72»72 clips altogether in April and May. #### 72
MATH	Mr. Madoff invests 1000 dollars in a fund that compounds annually at a constant interest rate. After three years, his investment has grown to 1225 dollars. What is the annual interest rate, as a percentage? (Round your answer to the nearest integer.)	Let r be the annual interest rate. Then after three years, Mr. Madoff's investment is $1000 \cdot \left(1+\frac{r}{100}\right)^3$, so $1000 \cdot \left(1+\frac{r}{100}\right)^3 = 1225$. Then $\left(1+\frac{r}{100}\right)^3 = 1.225$,so $[1+\frac{r}{100} = \sqrt[3]{1.225} = 1.069987\ldots$, which means $r = \boxed{7}$, to the nearest integer. #### 7.0

Table 5: Example of ground truth answers used for few-shot prompting.

Model	URL
Llama 3.1 70B	https://huggingface.co/meta-llama/Llama-3.1-70B-Instruct
Qwen 2.5 72B	https://huggingface.co/Qwen/Qwen2.5-72B-Instruct
Qwen 2.5 14B	https://huggingface.co/Qwen/Qwen2.5-14B-Instruct
Gemma 2 27B	https://huggingface.co/google/gemma-2-27b-it
Qwen 2.5 7B	https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
Gemma 2 9B	https://huggingface.co/google/gemma-1.1-9b-it
Llama 3.1 8B	https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct
Gemma 2 2B	https://huggingface.co/google/gemma-1.1-2b-it

Table 6: Used models and corresponding hyperlinks.

	GSM8K	AQUA-RAT	MATH
Llama 3.1 70B	93.25	78.57	47.11
Qwen 2.5 72B	**95.07**	**83.73**	**73.86**
Qwen 2.5 14B	93.48	82.54	64.47
Gemma 2 27B	85.97	67.46	38.80
Qwen 2.5 7B	88.10	75.40	60.31
Gemma 2 9B	80.52	61.51	31.31
Llama 3.1 8B	72.40	61.51	20.69
Gemma 2 2B	37.53	26.98	7.15

Table 7: Task performance of the investigated models.

B.2 Judging performance per model pair

We conduct experiments with all eight models serving as judges. We present the performance metrics of all judges across all model pairs in Figure 9.

C Examples

C.1 Example Subset Performance

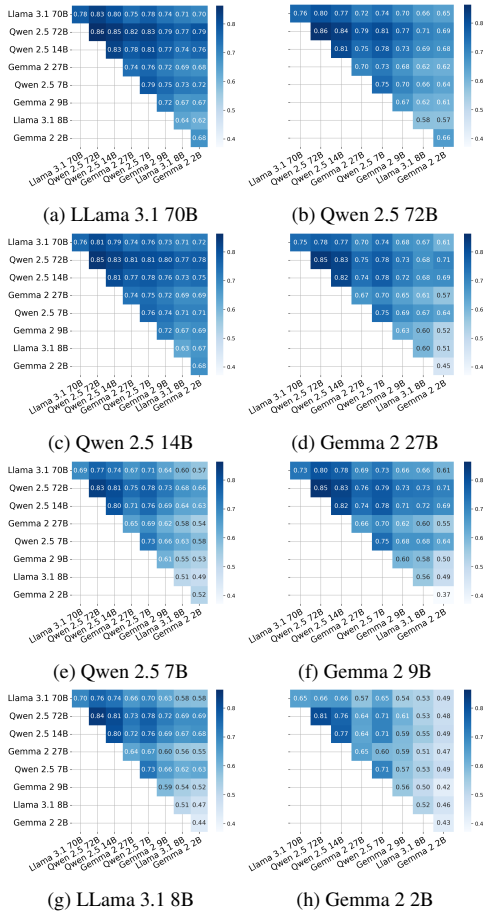
To better understand the correlation observed in Figure 4, we provide examples of these subsets, which can be seen in Table 8. These examples include the following details: the judge, the compared models, the dataset, the performance of the correct model on the dataset (denoted by S_A), the performance of the incorrect model on the dataset S_B, the judgment performance on the subset (denoted by $S^J_{A,B}$), and the size of the subset. We provide the three subsets with the highest performance, the three subsets with the lowest performance, and three random subsets where Llama 3.1 70B is the judge.

D Statistical Methodology

We describe the statistical background for the tests applied in Section 6. All predictions and statistical tests in Section 6 were performed using the statsmodels library (Seabold and Perktold, 2010).

Judge	model A	model B	dataset	S_A	S_B	$S^J_{A,B}$	No. Samples
Llama 3.1 70B	Qwen 2.5 14B	Gemma 2 2B	MATH	64.50	7.10	94.60	1655
Llama 3.1 70B	Qwen 2.5 72B	Gemma 2 2B	AQUA-RAT	83.70	27.00	94.50	309
Llama 3.1 70B	Qwen 2.5 7B	Gemma 2 2B	MATH	60.30	7.10	94.00	1520
Llama 3.1 70B	Qwen 2.5 72B	Qwen 2.5 7B	AQUA-RAT	83.70	75.40	62.50	64
Llama 3.1 70B	Qwen 2.5 7B	Qwen 2.5 14B	AQUA-RAT	75.40	82.50	42.30	26
Llama 3.1 70B	Qwen 2.5 72B	Qwen 2.5 72B	MATH	73.90	73.90	49.50	206
Llama 3.1 70B	Gemma 2 27B	Qwen 2.5 14B	AQUA-RAT	67.50	82.50	15.00	20
Llama 3.1 70B	Gemma 2 2B	Qwen 2.5 14B	GSM8K	37.50	93.50	15.00	20
Llama 3.1 70B	Gemma 2 2B	Llama 3.1 70B	MATH	7.10	47.10	14.50	62

Table 8: Examples of judgement performances on subsets where model A is correct and model B is incorrect.



(a) LLama 3.1 70B (b) Qwen 2.5 72B

(c) Qwen 2.5 14B (d) Gemma 2 27B

(e) Qwen 2.5 7B (f) Gemma 2 9B

(g) LLama 3.1 8B (h) Gemma 2 2B

Figure 9: Performance $S^J_{A,B}$ of LLM judges on model pairs, averaged across datasets.

D.1 Coefficient of Determination

The coefficient of determination, R^2, for evaluation of linear regression models (Fahrmeir et al., 2013) is defined as follows:

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

R^2 measures the share of the variance in Y explained by its covariation with the features $\mathbf{X}$ included in the model by dividing the variation of the *predicted* values $\hat{y}_i$ by the variation of the true target values y_i. If the features $\mathbf{X}$ have high explanatory power for Y, the $\hat{y}_i$ will be close to the y_i and R^2 will be close to 1, while in the extreme case of no correlation between $\mathbf{X}$ and Y the arithmetic mean is the best estimate (i.e., $\hat{y}_i = \bar{y} \; \forall \; i = 1, \ldots, n$) resulting in $R^2 = 0$.

D.2 Overall-F-Test

The Overall-F-Test is built upon R^2 and tests whether the overall model is of any significant value for explaining the variation of the target variable. The F-distributed test statistic is calculated as

$$\frac{R^2}{1 - R^2} \cdot \frac{n - p - 1}{p},$$

where R^2 is the coefficient of determination, n is the number of observations, and p is the number of covariates included in the model (i.e., the number of estimated coefficients excluding the intercept). The hypotheses that can be tested this way are

$$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$$

vs.

$$H_1 : \beta_j \neq 0 \text{ for at least one } j \in \{1, \ldots, p\}.$$

Model	Both correct	A correct	B correct	Both incorrect
Llama 3.1 70B	51.8	18.1	21.7	8.4
Qwen 2.5 72B	54.9	19.6	19.8	5.6
Qwen 2.5 14B	50.2	20.0	23.0	6.9
Gemma 2 27B	52.6	15.9	15.4	16.0

Table 9: Percentage of predictions individual models made.

So from a rejection of H_0, it can be concluded that at least one of the included features exhibits explanatory power for the variation of the target variable.

D.3 Multiple Testing

Since we conduct multiple statistical tests within the scope of one research project, it is important to consider multiple testing as a potential problem resulting in false positive findings. The p-values from our tests, however, also satisfy a significance level resulting from a Bonferroni Correction of the typical significance level of 5%.

E Sample-level Analysis

We utilize Scikit-learn (Pedregosa et al., 2011) library to train and evaluate the Logistic Regression and Random Forest Model. We use the standard settings for the Logistic Regression model. We use the Random Forests model with 1500 estimators, and standard settings apart from that.

During preprocessing, we use simple word splittling by spaces. We employ the english stop word removal integreated into Scikit-learn. We set the maximum number of features to 5,000, for the N-Gram of part-of-speech tags, we set the N-gram range from 5-grams up to 13-grams, following settings of Shaib et al. (2024). For training, we use the Scikit-learn (Pedregosa et al., 2011) library. The running time was negligible.

In Table 9