SBU Figures It Out: Models Explain Figurative Language

Mohadeseh Bastan*
Stony Brook University
mbastan@cs.stonybrook.edu

Yash Kumar Lal*
Stony Brook University
ylal@cs.stonybrook.edu

Abstract

Figurative language is ubiquitous in human communication. However, current NLP models are unable to demonstrate a significant understanding of instances of this phenomena. FigLang shared task on figurative language understanding posed the problem of predicting and explaining the relation between a premise and a hypothesis containing an instance of the use of figurative language. We experiment with different variations of using T5-large for this task and build a model that significantly outperforms the task baseline. Treating it as a new task for T5 and simply finetuning on the data achieves the best score on the defined evaluation. Furthermore, we find that hypothesis-only models are able to achieve most of the performance.

1 Introduction

Figurative language is an important component of discourse, ranging from daily interactions to books. It is used as a tool to convey complex and deeper emotions that are often difficult to express literally (Ghosh et al., 2015). Despite the fact that Transformer-based pretrained language models (LMs) get even larger, they are still unable to comprehend the physical world, cultural knowledge, or social context in which figurative language is embedded. Large-scale crowdsourced datasets often contain these phenomena inherently. To show true conceptual understanding of figurative language, the model should not only be able to correctly differentiate a figurative instance from its literal counterpart, but also explain its decision. These natural language explanations should be readily comprehensible by an end-user who needs to assert a model's reliability (Camburu et al., 2018; Wiegreffe and Marasovic, 2021).

This paper describes the experiments and submission of the LUNR lab at Stony Brook University, USA to the shared task on Figurative Language Understanding (Chakrabarty et al., 2022b) organized at EMNLP 2022. Given a premise and a hypothesis, the shared task required predicting the relation between them as well as an explanation for the same. We use variations in input format, separator and sequential fine-tuning techniques to build our final model.

Since the task involves predicting the label as well as an explanation for it, in this paper we vary the order of generation of each target in our models. Prior work (Khashabi et al., 2020) highlighted the importance of separator tokens. It helps the model distinguish between different portions of the input. Additionally, since this task is not a common one, variations in input format and keywords dictate how well a model performs. To that end, we experimented with different formats prescribed for T5 models as well as a simple one for an unseen, new task. Finally, we also experimented with sequential fine-tuning on several related datasets to improve performance on the shared task.

Our final model is a simple T5-large model fine-tuned on the task data, trained to generate the explanation before the label. The input format does not contain any task-specific keys and does not resemble any of the ones described in Raffel et al. (2020). The model uses a "\n" separator, which is a prominent part of how UnifiedQA (Khashabi et al., 2020) was built over T5. It improves significantly over the task baseline. We observe that (1) treating this as a new task leads to best model performance, (2) the dataset contains artifacts that hypothesis-only models use to reach significant performance, and (3) knowing the type of phenomena being encapsulated does not help the model.

2 Related Work

The model's ability to explain decisions has been investigated in previous studies. Rajani et al. (2019) presents a novel Common Sense Explanations

*First two authors have equal contribution

Proceedings of the 3rd Workshop on Figurative Language Processing (FLP), pages 143 - 149

(CoS-E) dataset to explore commonsense reasoning and propose a novel method, CAGE for automatically generating explanations that achieve state-of-the-art performance. Camburu et al. (2018) introduces a large corpus of human-annotated explanations for the Stanford Natural Language Inference (SNLI) (Bowman et al., 2015a) dataset which is collected to enable research in generation of free-form textual reasoning. Bastan et al. (2022) introduces SuMe dataset which generates relation between entities and an explanation for why this relation exists or how this relation comes about.

None of the previous work explored the possibility of different data formats. In this work we evaluate different combinations of the explanation and label generations. We also study the effect of the pretrained model on similar tasks as a sequential pretraining.

3 Data

The shared task data (Chakrabarty et al., 2022a) contains 9,000 high-quality literal, figurative sentence pairs with entail/contradict labels and the associated explanations. The benchmark spans five types of figurative language: Paraphrase, Sarcasm, Simile, Metaphor, and Idiom. The definition of each type is explained as follows:

Paraphrase is a rephrasing of something that is written. All sentences in this category belongs to the entailment category.

Sarcasm is using phrases which have the opposite meaning from what they are intended to convey. It can be used for creating contradiction labels.

Simile is using a figure of speech to compare something with something else. It can be used for both entailment and contradiction labels.

Metaphor is when a word or phrase used to describe something that it cannot literally describe. It can be used for both entailment and contradiction labels. It can be used for both entailment and contradiction labels.

Idiom is established by usage as having a meaning not derived from their individual meanings. It can be used for both entailment and contradiction labels.

A noteworthy property of this data is that both the entailment/contradiction labels and the explanations are w.r.t the figurative language expression (i.e., metaphor, simile, idiom) rather than other parts of the sentence. The task is challenging because it inherently requires 1) relational reasoning using background commonsense knowledge, and 2) finegrained understanding of figurative language.

We split 7,500 examples into a 80-20 train and dev set randomly. These sets are then used to build models for the overall shared task.

4 Experiment Design

We use the T5 (Raffel et al., 2020) family of models for our submission. Particularly, we build over T5-large.

Since this is a new task for T5, we experiment with various input and output formats. We build models where the label is placed before and after explanation on the target side. Large language models have also been shown to be sensitive to the choice of separators. To this end, we build models that conform to different input/output formats as well as separators.

Prior work has shown that pretraining on large amounts of data similar to the task improves the downstream performance of models. To this end, we use e-SNLI (Camburu et al., 2018) to sequential fine-tuning our model before finetuning on downstream task data to obtain a final model. e-SNLI is an extension of the SNLI dataset (Bowman et al., 2015b) with an additional layer of human-annotated natural language explanations of the entailment relations. Similarly, SuMe Bastan et al. (2022) is a biomedical mechanism explanation dataset which contains a set of supporting sentence about two main entities, the relation between the entities, and a sentence explaining the mechanism behind this relation. They explored the generation of explanation and target label at the same time given the supporting sentences, using different transformer based models. They use [explanation. label] as the output format while we explore all possible orders and separator tokens. We used the model pretrained on SuMe dataset and finetuned on this task.

Poliak et al. (2018) used hypothesis-only models showed that statistical irregularities may allow a model to perform natural language inference in some datasets beyond what should be achievable without access to the context. Motivated by that, we also build hypothesis-only models to analyze whether models require contexts to perform this NLI + explanation task.

5 Results

5.1 Evaluation

To evaluate the performance of each model, we use two generations and three classifications metrics. For generations, we use BLEURT (Sellam et al., 2020) and BERTScore (Zhang et al., 2019) which have been proven to be more effective than tradition ROUGE scores. In order to evaluate the quality of explanations, we compute the average between these two scores. NLI label accuracy is then reported based on three explanation average score thresholds. We compute the accuracy@0 meaning accuracy on all generated data, accuracy@50 meaning accuracy of the generated label for all texts with average explanation score higher than 50, and accuracy@60 which is the accuracy of the generated label for all texts with average explanation score higher than 60. This evaluation scheme has been defined by the task organizers themselves.

5.2 Task Results

The baseline model released for the task is T5-3B finetuned on this dataset (Chakrabarty et al., 2022b). Our best model is a T5-large finetuned on task data in using RTE keywords with the "[SEP]" separator, and predicting the label before the explanation. It significantly improves upon the baseline set for the shared task despite being much smaller in terms of number of parameters. Particularly, we observe that Acc@60 is much lower than Acc@50, which means that the average accuracy of the generated label drops as the average explanation score threshold goes up from 50 to 60 (becomes stricter).

	Acc@0	Acc@50	Acc@60
Our Model	**0.889**	**0.824**	**0.517**
Baseline	0.767	0.691	0.443

Table 1: Results on shared task test set

6 Analysis

We analyse the performance of the numerous models that we have built to understand the impact of various design decisions that we took — input format, sequential fine-tuning, and order of required predictions. Further, we also want to understand the impact of artifacts present in the dataset itself on model performance. We use the evaluation described in subsection 5.1 on the dev set for analysis.

6.1 How does input format affect performance?

The data input formats vary in two aspects — task-specific keywords and the separator. Specifically, the task-specific keywords can correspond to a new task for T5 (no keywords), RTE and MNLI (Appendix D.2 and D.3 of Raffel et al. (2020) respectively). We experiment with three possible separators between pieces of input text - ‘ ’ (whitespace), [SEP] (the sep token), and "\n" (the newline character). Both **\n** and **[SEP]** are predefined to the tokenizer as one unique token before training.

The effects of these design choices can be seen in Table 2. We find that treating this as a new task (and not using any predefined task-specific keywords) yields the best model performance. Furthermore, predicting the label before predicting its explanation is better than the opposite. This is in line with the expected order of performing both tasks — one would predict the relation between the pair before explaining it. We also see that using the [SEP] token is better for the label before explanation setting except when using the MNLI task format.

6.2 Does sequential fine-tuning help?

Prior work has shown that sequential fine-tuning on similar tasks often helps models. Both e-SNLI (Camburu et al., 2018) and SuMe (Bastan et al., 2022) are tasks where models have to predict labels as well as explain it. We built models that were first psequential fine-tuning on one of these datasets and then finetuned on the task data. The results of these two experiments are shown in Table 3.

We found that the sequential fine-tuning paradigm actually hurts model performance significantly, no matter which task is used with the model first. We hypothesize that while these selected tasks are similar in terms of what the model has to predict, they do not capture any aspects of the figurative language phenomena. So, introducing a model to these tasks does not necessarily nudge it towards the right domain.

6.3 Label before explanation vs explanation before label

We explored different order of generation for the label and the explanation. First, for each data, we set the label to be generated before the explanation (*lbe*) then we changed the order and first generated the explanation before the label (*ebl*).

The results are shown in Table 2. We find that

Keyword	Label Position	Seperator	Model Name	Acc@0	Acc@50	Acc@60
-	after	-	ebl-no	0.830	0.778	0.557
-	after	[SEP]	ebl-sep	0.789	0.737	0.513
-	after	\n	ebl-slashn	0.822	0.766	0.557
-	before	-	lbe-no	0.838	0.773	0.531
-	before	[SEP]	lbe-sep	**0.899**	**0.830**	0.584
-	before	\n	be-slashn	0.844	0.789	0.539
mnli	after	-	mnli-ebl-no	0.790	0.737	0.514
mnli	after	[SEP]	mnli-ebl-sep	0.779	0.721	0.512
mnli	after	\n	mnli-ebl-slashn	0.814	0.747	0.540
mnli	before	-	mnli-lbe-no	0.799	0.754	0.537
mnli	before	[SEP]	mnli-lbe-sep	0.711	0.672	0.451
mnli	before	\n	mnli-lbe-slashn	0.788	0.738	0.529
rte	after	-	rte-ebl-no	0.737	0.690	0.486
rte	after	[SEP]	rte-ebl-sep	0.797	0.748	0.537
rte	after	\n	rte-ebl-slashn	0.833	0.767	0.531
rte	before	-	rte-lbe-no	0.797	0.745	0.510
rte	before	[SEP]	rte-lbe-sep	0.891	0.827	**0.590**
rte	before	\n	rte-lbe-slashn	0.741	0.698	0.476

Table 2: Model performance with different input formats on the dev set. The first column shows the task specific keyword we used in finetuning. It's either nothing, the same as 'mnli' task, or 'rte' task. The second column indicates whether the label is generated before or after the explanation. *lbe* means that the model was trained to generate the label before the explanation while *ebl* indicates the opposite. The third column indicates which separator was used between the label and the explanation. We either used no token, [SEP] token, or \n token. Model name comes from the combination of the previous three columns. This notation is used in all other tables as well. Treating the shared task as a new T5 task, using the [SEP] token as separator, and predicting the label before the explanation helps us build the best model.

Model Name	Acc@0	Acc@50	Acc@60
lbe-sep	**0.899**	**0.830**	**0.584**
esnli-mnli-ebl-sep	0.73	0.666	0.413
sume-mnli-ebl-slashn	0.696	0.672	0.502
sume-mnli-lbe-sep	0.729	0.669	0.410

Table 3: Effect of sequential fine-tuning on model performance on shared task data. We only include the best possible model scores obtained in the no-, esnli- and sume- sequential fine-tuning regime. Clearly, sequential fine-tuning only has a negative impact on model performance.

predicting the label before moving on to the explanation is better for the model in both a new task and the RTE task setup. However, the opposite is true for MNLI. Why the pattern does not hold remains an open research issue.

6.4 Presence of artifacts in the dataset

Poliak et al. (2018) showed the presence of artifacts in several popular NLI datasets. We use a similar approach and build hypothesis-only models to test the presence of artifacts in this dataset and task. Ideally, these models should perform very poorly on this data since they do not have access to the premise and have to judge incomplete inputs.

Model Name	Acc@0	Acc@50	Acc@60
lbe-sep	0.672	0.627	**0.423**
mnli-lbe-no	**0.696**	**0.634**	0.418
rte-lbe-no	0.680	0.622	0.416

Table 4: Performance of hypothesis-only models on the task. The table only includes the best performing model from each input format task type (new, mnli and rte).

Table 4 shows that models are able to achieve high enough Acc@0 scores, showing that the overall dataset contains some artifacts. Technically, if a significant portion of the dataset can be correctly classified without looking at the premise (well beyond the most-frequent-class baseline), it shows that it is possible to perform well on the datasets

Model Name	Acc@0	Acc@50	Acc@60
ebl-no	0.854	0.803	0.542
mnli-ebl-no	0.847	0.786	0.567
rte-ebl-no	**0.862**	**0.804**	**0.584**

Table 5: Performance of models when they are also provided the type of phenomena captured in the premise-hypothesis. We only include the best performing model from each input format task type (new, mnli and rte).

without modeling natural language inference hence the data relies on annotation artifacts (Gururangan et al., 2018). However, it is also clear that using Acc@60 shows the weakness of the explanations generated by these models. Overall, we posit that using hypothesis-only models alone are also effective in performing this task.

6.5 Does knowing the type of figurative language phenomena help?

Wang et al. (2019) showed that additional knowledge is useful in improving NLI models. The dataset is annotated with the type of figurative phenomena encapsulated in the premise-hypothesis pair. Using this additional information can help a model predict the relation between the pair better, and nudge it towards the correct explanation.

Performance for such models is listed in Table 5. We find that knowing the type of phenomena hurts the model as compared to just simply finetuning with vanilla task inputs and outputs. It is unclear why this additional knowledge has a negative impact. One assumption can be because this additional information is not available at the test data, we can only use this information during training. This study is done on the development set. We trained a model with this additional information, but at the time of evaluation we didn't use this as this is not available in the test set.

7 Conclusion

Figurative language is an important component of discourse, often used as a tool to convey complex emotions usually difficult to express literally. The shared task is designed to test whether models can predict the relation between a pair of sentences that contains figurative language as well as explain that phenomena. We experiment with building several models based on T5-large varying the input format, order of prediction and sequential fine-tuning.

Our final model is a simple T5-large model fine-tuned on the task data, trained to generate the explanation before the label. The input format does not contain any task-specific keys and does not resemble any of the ones described in Raffel et al. (2020) but uses a "\n" separator. It improves significantly over the task baseline. We observe that (1) treating this as a new task leads to best model performance, (2) the dataset contains artifacts that hypothesis-only models use to reach significant performance, and (3) knowing the type of phenomena being encapsulated does not help the model.

8 Limitations

Our approach is fundamentally limited by the limits of the fine-tuned transformer based models since we only used one specific t5-large model. Further, it might be computationally prohibitive to try larger models since in requires more resources and computational machines. We focus on exploring different preprocessing steps, whereas a significant amount of errors stem from the capacity of the model in generating good explanations.

References

Mohaddeseh Bastan, Nishant Shankar, Mihai Surdeanu, and Niranjan Balasubramanian. 2022. Sume: A dataset towards summarizing biomedical mechanisms. *arXiv preprint arXiv:2205.04652*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015a. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015b. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 9539–9549. Curran Associates, Inc.

Tuhin Chakrabarty, A. Saakyan, Debanjan Ghosh, and Smaranda Muresan. 2022a. Flute: Figurative language understanding through textual explanations.

Tuhin Chakrabarty, Arkadiy Saakyan, Debanjan Ghosh, and Smaranda Muresan. 2022b. Flute: Figurative language understanding and textual explanations. *arXiv preprint arXiv:2205.12404*.

Aniruddha Ghosh, Guofu Li, Tony Veale, Paolo Rosso, Ekaterina Shutova, John Barnden, and Antonio Reyes. 2015. Semeval-2015 task 11: Sentiment analysis of figurative language in twitter. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 470–478.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur P Parikh. 2020. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*.

Xiaoyan Wang, Pavan Kapanipathi, Ryan Musa, Mo Yu, Kartik Talamadupula, Ibrahim Abdelaziz, Maria Chang, Achille Fokoue, Bassem Makni, Nicholas Mattei, et al. 2019. Improving natural language inference using external knowledge in the science questions domain. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7208–7215.

Sarah Wiegreffe and Ana Marasovic. 2021. Teach me to explain: A review of datasets for explainable natural language processing. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Sequential Fine-tuning

The extended results of the model pretrained on SuMe (Bastan et al., 2022) is shown in Table 6 and the results of the model pretrained on e-snli (Camburu et al., 2018) and fine-tuned on this task is shown in Table 7. Since the sequential fine-tuning on esnli is time and resource consuming, we only explored a few set of preprocessing on this task.

Model Name	Acc@0	Acc@50	Acc@60
ebl-no	0.606	0.541	0.324
ebl-sep	0.653	0.593	0.379
ebl-slashn	0.648	0.624	0.455
lbe-no	0.674	0.615	0.375
lbe-sep	0.696	0.637	0.388
lbe-slashn	0.687	0.655	0.460
mnli-ebl-no	0.684	0.628	0.408
mnli-ebl-sep	0.701	0.641	0.410
mnli-ebl-slashn	0.696	**0.672**	**0.502**
mnli-lbe-no	0.714	0.643	0.394
mnli-lbe-sep	**0.729**	0.669	0.410
mnli-lbe-slashn	0.689	0.661	0.488
rte-ebl-no	0.676	0.625	0.402
rte-ebl-sep	0.691	0.604	0.347
rte-ebl-slashn	0.680	0.662	0.488
rte-lbe-no	0.713	0.652	0.402
rte-lbe-sep	0.701	0.643	0.397
rte-lbe-slashn	0.682	0.657	0.472

Table 6: SuMe Pretrained Models Performance

Model Name	Acc@0	Acc@50	Acc@60
ebl-no	0.727	0.655	0.375
mnli-ebl-sep	**0.73**	**0.666**	**0.413**

Table 7: ESNLI Pretrained Models Performance

B Hypothesis-only Models

The hypothesis-only experiments show the presence of artifacts in this dataset. The full performance of these models are shown in Table 4.

C Effect of Knowing the Phenomena

The extended results of the model with the *type* information is shown in Table 5.

Model Name	Acc@0	Acc@50	Acc@60
ebl-no	0.638	0.578	0.363
ebl-sep	0.637	0.576	0.381
ebl-slashn	0.676	0.612	0.404
lbe-no	0.684	0.604	0.410
lbe-sep	0.672	0.627	**0.423**
lbe-slashn	0.672	0.611	0.398
mnli-ebl-no	0.639	0.563	0.362
mnli-ebl-sep	0.670	0.596	0.378
mnli-ebl-slashn	0.661	0.593	0.390
mnli-lbe-no	**0.696**	**0.634**	0.418
mnli-lbe-sep	0.676	0.618	0.411
mnli-lbe-slashn	0.674	0.603	0.402
rte-ebl-no	0.632	0.569	0.366
rte-ebl-sep	0.637	0.574	0.363
rte-ebl-slashn	0.634	0.561	0.351
rte-lbe-no	0.680	0.622	0.416
rte-lbe-sep	0.678	0.628	0.398
rte-lbe-slashn	0.679	0.607	0.409

Table 8: Hypothesis Only Performance

Model Name	Acc@0	Acc@50	Acc@60
ebl-no	0.854	0.803	0.542
ebl-sep	0.826	0.766	0.520
ebl-slashn	0.839	0.780	0.543
lbe-no	0.754	0.712	0.490
lbe-sep	0.742	0.690	0.488
lbe-slashn	0.740	0.694	0.496
mnli-ebl-no	0.847	0.786	**0.567**
mnli-ebl-sep	0.834	0.776	0.560
mnli-ebl-slashn	0.819	0.771	0.528
mnli-lbe-no	0.741	0.694	0.503
mnli-lbe-sep	0.756	0.709	0.487
mnli-lbe-slashn	0.755	0.713	0.509
rte-ebl-no	**0.862**	**0.804**	0.584
rte-ebl-sep	0.816	0.786	0.536
rte-ebl-slashn	0.821	0.775	0.533
rte-lbe-no	0.738	0.705	0.485
rte-lbe-sep	0.762	0.713	0.525
rte-lbe-slashn	0.758	0.719	0.509

Table 9: Type Added Performance