ExpBERT: Representation Engineering with Natural Language Explanations

Shikhar Murty **Pang Wei Koh** **Percy Liang**
Computer Science Department, Stanford University
{smurty,pangwei,pliang}@cs.stanford.edu

Abstract

Suppose we want to specify the inductive bias that married couples typically go on honeymoons for the task of extracting pairs of spouses from text. In this paper, we allow model developers to specify these types of inductive biases as natural language explanations. We use BERT fine-tuned on MultiNLI to "interpret" these explanations with respect to the input sentence, producing explanation-guided representations of the input. Across three relation extraction tasks, our method, ExpBERT, matches a BERT baseline but with 3–20× less labeled data and improves on the baseline by 3–10 F1 points with the same amount of labeled data.

1 Introduction

Consider the relation extraction task of finding spouses in text, and suppose we wanted to specify the inductive bias that married couples typically go on honeymoons. In a traditional feature engineering approach, we might try to construct a "did they go on a honeymoon?" feature and add that to the model. In a modern neural network setting, however, it is not obvious how to use standard approaches like careful neural architecture design or data augmentation to induce such an inductive bias. In a way, while the shift from feature engineering towards end-to-end neural networks and representation learning has alleviated the burden of manual feature engineering and increased model expressivity, it has also reduced our control over the inductive biases of a model.

In this paper, we explore using natural language explanations (Figure 1) to generate features that can augment modern neural representations. This imbues representations with inductive biases corresponding to the explanations, thereby restoring some degree of control while maintaining their expressive power.

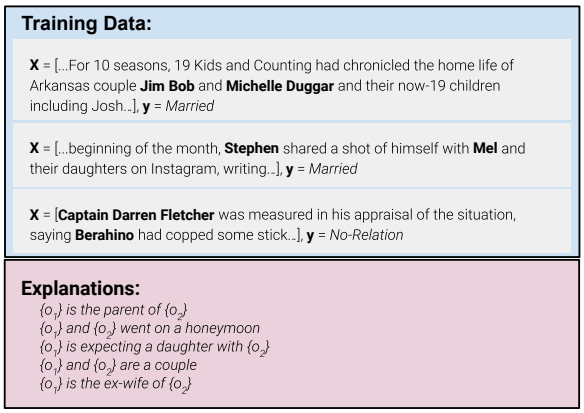

Figure 1: Sample data points and explanations from Spouse, one of our relation extraction tasks. The explanations provide relevant features for classification.

Prior work on training models with explanations use semantic parsers to interpret explanations: the parser converts each explanation into an executable logical form that is executable over the input sentence and uses the resulting outputs as features (Srivastava et al., 2017) or as noisy labels on unlabeled data (Hancock et al., 2018). However, semantic parsers can typically only parse low-level statements like "*'wife' appears between* $\{o_1\}$ *and* $\{o_2\}$ *and the last word of* $\{o_1\}$ *is the same as the last word of* $\{o_2\}$" (Hancock et al., 2018).

We remove these limitations by using modern distributed language representations, instead of semantic parsers, to interpret language explanations. Our approach, ExpBERT (Figure 2), uses BERT (Devlin et al., 2019) fine-tuned on the MultiNLI natural language inference dataset (Williams et al., 2018) to produce features that "interpret" each explanation on an input. We then use these features to augment the input representation. Just as a semantic parser grounds an explanation by converting it into a logical form and then executing it, the features produced by BERT can be seen as a soft "execution" of the explanation on the input.

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 2106–2113
July 5 - 10, 2020.

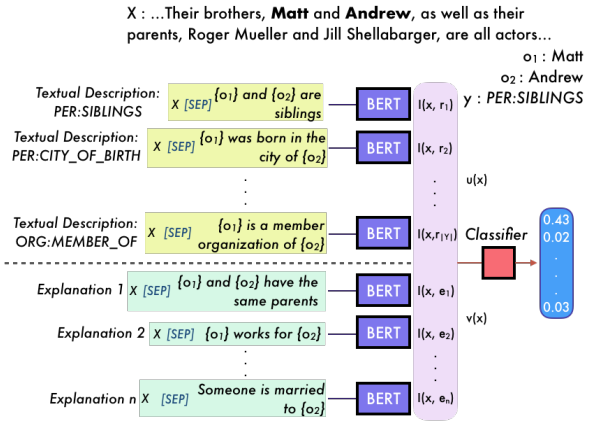

Figure 2: Overview of our approach. Explanations as well as textual descriptions of relations are interpreted using BERT for a given x to produce a representation which form inputs to our classifier.

On three benchmark relation extraction tasks, ExpBERT improves over a BERT baseline with no explanations: it achieves an F1 score of 3–10 points higher with the same amount of labeled data, and a similar F1 score as the full-data baseline but with 3–20x less labeled data. ExpBERT also improves on a semantic parsing baseline (+3 to 5 points F1), suggesting that natural language explanations can be richer than low-level, programmatic explanations.

2 Setup

Problem. We consider the task of relation extraction: Given $x = (s, o_1, o_2)$, where s is a sequence of words and o_1 and o_2 are two entities that are substrings within s, our goal is to classify the relation $y \in \mathcal{Y}$ between o_1 and o_2. The label space $\mathcal{Y}$ includes a NO-RELATION label if no relation applies. Additionally, we are given a set of natural language explanations $\mathcal{E} = \{e_1, e_2, \ldots, e_n\}$ designed to capture relevant features of the input for classification. These explanations are used to define a *global* collection of features and are not tied to individual examples.

Approach. Our approach (Figure 2) uses pretrained neural models to interpret the explanations $\mathcal{E}$ in the context of a given input x. Formally, we define an *interpreter* $\mathcal{I}$ as any function that takes an input x and explanation e_j and produces a feature vector in $\mathbf{R}^d$. In our ExpBERT implementation, we choose $\mathcal{I}$ to capture whether the explanation e_j is entailed by the input x. Concretely, we use BERT (Devlin et al., 2019) fine-tuned on MultiNLI (Williams et al., 2018): we feed wordpiece-tokenized versions of the explanation e_j (hypothesis) and the instance x (premise), separated by a [SEP] token, to BERT. Following standard practice, we use the vector at the [CLS] token to represent the entire input as a 768-dimensional feature vector:

$$\mathcal{I}(x, e_j) = \text{BERT}\left([\text{CLS}], s, [\text{SEP}], e_j\right). \quad (1)$$

These vectors, one for each of the n explanations, are concatenated to form the *explanation representation* $v(x) \in \mathbf{R}^{768n}$,

$$v(x) = \left[\mathcal{I}(x, e_1), \mathcal{I}(x, e_2), \ldots, \mathcal{I}(x, e_n)\right]. \quad (2)$$

In addition to $v(x)$, we also map x into an *input representation* $u(x) \in \mathbf{R}^{768|\mathcal{Y}|}$ by using the same interpreter over textual descriptions of each potential relation. Specifically, we map each potential relation y_i in the label space $\mathcal{Y}$ to a textual description r_i (Figure 2), apply $\mathcal{I}(x, \cdot)$ to r_i, and concatenate the resulting feature vectors:

$$u(x) = \left[\mathcal{I}(x, r_1), \mathcal{I}(x, r_2), \ldots, \mathcal{I}(x, r_{|\mathcal{Y}|})\right]. \quad (3)$$

Finally, we train a classifier over $u(x)$ and $v(x)$:

$$f_\theta(x) = \text{MLP}\left[u(x), v(x)\right]. \quad (4)$$

Note that $u(x)$ and $v(x)$ can be obtained in a preprocessing step since $\mathcal{I}(\cdot, \cdot)$ is fixed (i.e., we do not additionally fine-tune BERT on our tasks). For more model details, please refer to Appendix A.1.

Baselines. We compare ExpBERT against several baselines that train a classifier over the same input representation $u(x)$. **NoExp** trains a classifier only on $u(x)$. The other baselines augment $u(x)$ with variants of the explanation representation $v(x)$. **BERT+SemParser** uses the semantic parser from Hancock et al. (2018) to convert explanations into executable logical forms. The resulting denotations over the input x (a single bit for each explanation) are used as the explanation representation, i.e., $v(x) \in \{0, 1\}^n$. We use two different sets of explanations for this baseline: our natural language explanations (LangExp) and the low-level explanations from Hancock et al. (2018) that are more suitable for the semantic parser (ProgExp). **BERT+Patterns** converts explanations into a collection of unigram, bigram, and trigram patterns and creates a binary feature for each pattern based on whether it is contained in s or not. This gives $v(x) \in \{0, 1\}^{n'}$, where n' is the number of patterns. Finally, we compare ExpBERT against a

Table 1: Dataset statistics.

Dataset	Train	Val	Test	Explanations
Spouse	22055	2784	2680	40
Disease	6667	773	4101	28
TACRED	68124	22631	15509	128

variant called **ExpBERT-Prob**, where we directly use entailment probabilities obtained by BERT (instead of the feature vector at the [CLS] token) as the explanation representation $v(x) \in [0,1]^n$.

3 Experiments

Datasets. We consider 3 relation extraction datasets from various domains—Spouse and Disease (Hancock et al., 2018), and TACRED (Zhang et al., 2017). Spouse involves classifying if two entities are married; Disease involves classifying whether the first entity (a chemical) is a cause of the second entity (a disease); and TACRED involves classifying the relation between the two entities into one of 41 categories. Dataset statistics are in Table 1; for more details, see Appendix A.2.

Explanations. To construct explanations, we randomly sampled 50 training examples for each $y \in \mathcal{Y}$ and wrote a collection of natural language statements explaining the gold label for each example. For Spouse and Disease, we additionally wrote some negative explanations for the NO-RELATION category. To interpret explanations for Disease, we use SciBERT, a variant of BERT that is better suited for scientific text (Beltagy et al., 2019). A list of explanations can be found in Appendix A.3.

Benchmarks. We find that explanations improve model performance across all three datasets: ExpBERT improves on the NoExp baseline by +10.6 F1 points on Spouse, +2.7 points on Disease, and +3.2 points on TACRED (Table 2).[1] On TACRED, which is the most well-established of our benchmarks and on which there is significant prior work, ExpBERT (which uses a smaller BERT-base model that is not fine-tuned on our task) outperforms the standard, fine-tuned BERT-large model by +1.5 F1 points (Joshi et al., 2019). Prior work on Spouse and Disease used a simple logistic classifier over traditional features created from dependency paths of the input sentence. This performs poorly compared to neural models, and our models attain significantly higher accuracies (Hancock et al., 2018).

Using BERT to interpret natural language explanations improves on using semantic parsers to evaluate programmatic explanations (+5.5 and +2.7 over BERT+SemParser (ProgExp) on Spouse and Disease, respectively). ExpBERT also outperforms the BERT+SemParser (LangExp) model by +9.9 and +3.3 points on Spouse and Disease. We exclude these results on TACRED as it was not studied in Hancock et al. (2018), so we did not have a corresponding semantic parser and set of programmatic explanations.

We note that ExpBERT—which uses the full 768-dimensional feature vector from each explanation—outperforms ExpBERT (Prob), which summarizes these vectors into one number per explanation, by +2–5 F1 points across all three datasets.

Data efficiency. Collecting a set of explanations $\mathcal{E}$ requires additional effort—it took the authors about 1 minute or less to construct each explanation, though we note that it only needs to be done once per dataset (not per example). However, collecting a small number of explanations can significantly and disproportionately reduce the number of labeled examples required. We trained ExpBERT and the NoExp baseline with varying fractions of Spouse and TACRED training data (Figure 3). ExpBERT matches the NoExp baseline with 20x less data on Spouse; i.e., we obtain the same performance with ExpBERT with 40 explanations and 2k labeled training examples as with NoExp with 22k examples. On TACRED, ExpBERT requires 3x less data, obtaining the same performance with 128 explanations and 23k training examples as compared to NoExp with 68k examples. These results suggest that the higher-bandwidth signal in language can help models be more data-efficient.

4 Analysis

4.1 Which explanations are important?

To understand which explanations are important, we group explanations into a few semantic categories (details in Appendix A.3) and cumulatively add them to the NoExp baseline. In particular, we break down explanations for Spouse into the

[1] We measure performance using F1 scores due to the class imbalance in the datasets (Spouse: 8% positive, Disease: 20.8% positive, and TACRED: 20.5% examples with a relation).

Table 2: Results on relation extraction datasets. For Spouse and Disease, we report 95% confidence intervals and for TACRED, we follow the evaluation protocol from Zhang et al. (2017). More details in Appendix A.

Model	Spouse	Disease	TACRED
NoExp	52.9 ± 0.97	49.7 ± 1.01	64.7
BERT+Patterns	53.3 ± 1.24	49.0 ± 1.15	64.4
BERT+SemParse (LangExp)	53.6 ± 0.38	49.1 ± 0.47	-
BERT+SemParse (ProgExp)	58.3 ± 1.10	49.7 ± 0.54	-
ExpBERT-Prob	58.4 ± 1.22	49.7 ± 1.21	65.3
ExpBERT	**63.5 ± 1.40**	**52.4 ± 1.23**	**67.9**

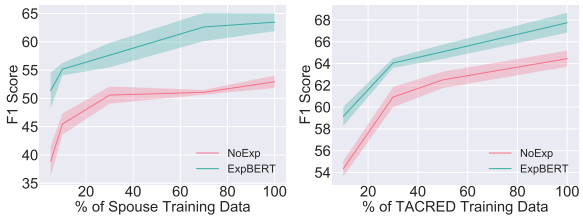

Figure 3: ExpBERT matches the performance of the NoExp baseline with 20x less data on Spouse (Left), and with 3x less data on TACRED (Right).

Table 3: Importance of various explanation groups.

Model	Spouse
NoExp	52.9 ± 0.97
+ MARRIED	55.2 ± 0.43
+ CHILDREN	55.9 ± 0.98
+ ENGAGED	57.0 ± 2.57
+ NEGATIVES	60.1 ± 0.87
+ MISC (full ExpBERT)	63.5 ± 1.40

groups MARRIED (10 explanations), CHILDREN (5 explanations), ENGAGED (3 explanations), NEGATIVES (13 explanations) and MISC (9 explanations). We find that adding new explanation groups helps performance (Table 3), which suggests that a broad coverage of various explanatory factors could be helpful for performance. We also observe that the MARRIED group (which contains paraphrases of $\{o_1\}$ *is married to* $\{o_2\}$) alone boosts performance over NoExp, which suggests that a variety of paraphrases of the same explanation can improve performance.

4.2 Quality vs. quantity of explanations

We now test whether ExpBERT can do equally well with the same number of *random* explanations, obtained by replacing words in the explanation with random words. The results are dataset-specific: random explanations help on Spouse but not on Disease. However, in both cases, random explanations do significantly worse than the original explanations (Table 4). Separately adding 10 random explanations to our original explanations led to a slight drop (≈1 F1 point) in accuracy. These results suggest that ExpBERT's performance comes from having a diverse set of high quality explanations and are not just due to providing more features.

Table 4: ExpBERT accuracy is significantly lower when we replace words in the original explanations with random words.

Model	Spouse	Disease
NoExp	52.9 ± 0.97	49.7 ± 1.01
ExpBERT (random)	56.4 ± 1.20	49.6 ± 1.22
ExpBERT (orig)	63.5 ± 1.40	52.4 ± 1.23
ExpBERT (orig + random)	62.4 ± 1.41	51.8 ± 1.03

Table 5: Combining language explanations with the external CTD ontology improves accuracy on Disease.

Model	Disease
ExpBERT	52.4 ± 1.23
ExpBERT (+ External)	59.1 ± 3.26

4.3 Complementing language explanations with external databases

Natural language explanations can capture different types of inductive biases and prior knowledge, but some types of prior knowledge are of course better introduced through other means. We wrap up our experiments with a vignette on how language explanations can complement other forms of feature and representation engineering. We consider Disease, where we have access to an external ontology (Comparative Toxicogenomic Database or CTD) from Wei et al. (2015) containing chemical-disease interactions. Following Hancock et al. (2018), we add 6 bits to the explanation representation $v(x)$ that test if the given chemical-disease pair follows certain relations in CTD (e.g., if they are in the ctd-therapy dictionary). Table 5 shows that as expected, other sources of information can complement language explanations in ExpBERT.

5 Related work

Many other works have used language to guide model training. As mentioned above, semantic parsers have been used to convert language explanations into features (Srivastava et al., 2017) and noisy labels on unlabeled data (Hancock et al., 2018; Wang et al., 2019).

Rather than using language to define a global collection of features, Rajani et al. (2019) and Camburu et al. (2018) use *instance*-level explanations to train models that generate their own explanations. Zaidan and Eisner (2008) ask annotators to highlight important words, then learn a generative model over parameters given these rationales. Others have also used language to directly produce parameters of a classifier (Ba et al., 2015) and as part of the parameter space of a classifier (Andreas et al., 2017).

While the above works consider learning from static language supervision, Li et al. (2016) and Weston (2016) learn from language supervision in an interactive setting. In a related line of work, Wang et al. (2017), users teach a system high-level concepts via language.

6 Discussion

Recent progress in general-purpose language representation models like BERT open up new opportunities to incorporate language into learning. In this work, we show how using these models with natural language explanations can allow us to leverage a richer set of explanations than if we were constrained to only use explanations that can be programmatically evaluated, e.g., through n-gram matching (BERT+Patterns) or semantic parsing (BERT+SemParser).

The ability to incorporate prior knowledge of the "right" inductive biases into model representations dangles the prospect of building models that are more robust. However, more work will need to be done to make this approach more broadly applicable. We outline two such avenues of future work. First, combining our ExpBERT approach with more complex state-of-the-art models can be conceptually straightforward (e.g., we could swap out BERT-base for a larger model) but can sometimes also require overcoming technical hurdles. For example, we do not fine-tune ExpBERT in this paper; doing so might boost performance, but fine-tuning through all of the explanations on each example is computationally intensive.

Second, in this paper we provided a proof-of-concept for several relation extraction tasks, relying on the fact that models trained on existing natural language inference datasets (like MultiNLI) could be applied directly to the input sentence and explanation pair. Extending ExpBERT to other natural language tasks where this relationship might not hold is an open problem that would entail finding different ways of interpreting an explanation with respect to the input.

Acknowledgements

We are grateful to Robin Jia, Peng Qi, John Hewitt, Amita Kamath, and other members of the Stanford NLP Group for helpful discussions and suggestions. We also thank Yuhao Zhang for assistance with TACRED experiments. PWK was supported by the Facebook Fellowship Program. Toyota Research Institute (TRI) provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

Reproducibility

Code and model checkpoints are available at https://github.com/MurtyShikhar/ExpBERT. The features generated by various interpreters can also be found at that link.

References

Jacob Andreas, Dan Klein, and Sergey Levine. 2017. Learning with latent language. In *NAACL-HLT*.

Jimmy Ba, Kevin Swersky, Sanja Fidler, and Ruslan Salakhutdinov. 2015. Predicting deep zero-shot convolutional neural networks using textual descriptions. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4247–4255.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. Scibert: Pretrained language model for scientific text. In *EMNLP*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. e-snli: Natural language inference with natural language explanations. In *Advances in Neural Information Processing Systems 31*, pages 9539–9549. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, and Christopher Ré. 2018. Training classifiers with natural language explanations. *Proceedings of the conference. Association for Computational Linguistics. Meeting*, 2018:1884–1895.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. 2019. Spanbert: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Jiwei Li, Alexander H. Miller, Sumit Chopra, Marc'Aurelio Ranzato, and Jason Weston. 2016. Learning through dialogue interactions by asking questions. In *ICLR*.

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.

Shashank Srivastava, Igor Labutov, and Tom Mitchell. 2017. Joint concept learning and semantic parsing from natural language explanations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1527–1536, Copenhagen, Denmark. Association for Computational Linguistics.

Sida I Wang, Samuel Ginn, Percy Liang, and Christopher D Manning. 2017. Naturalizing a programming language via interactive learning. In *55th Annual Meeting of the Association for Computational Linguistics, ACL 2017*, pages 929–938. Association for Computational Linguistics (ACL).

Ziqi Wang, Yujia Qin, Wenxuan Zhou, Jun Yan, Qinyuan Ye, Leonardo Neves, Zhiyuan Liu, and Xiang Ren. 2019. Learning to annotate: Modularizing data augmentation for textclassifiers with natural language explanations. *arXiv preprint arXiv:1911.01352*.

Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wiegers, and Zhiyong Lu. 2015. Overview of the biocreative v chemical disease relation (cdr) task.

Jason Weston. 2016. Dialog-based language learning. In *NIPS*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Omar Zaidan and Jason Eisner. 2008. Modeling annotators: A generative approach to learning from annotator rationales. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 31–40, Honolulu, Hawaii. Association for Computational Linguistics.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, pages 35–45.

A Appendix

A.1 Implementation Details

Interpreting explanations. When interpreting an explanation e_i on a particular example $x = (s, o_1, o_2)$, we first substitute o_1 and o_2 into the placeholders in the explanation e_i to produce an instance-level version of the explanation. For example, "$\{o_1\}$ and $\{o_2\}$ are a couple" might become "Jim Bob and Michelle Duggar are a couple".

Model hyperparameters and evaluation. We use BERT-BASE-UNCASED for `Spouse` and `TACRED`, and SCIBERT-SCIVOCAB-UNCASED for `Disease` from Beltagy et al. (2019). We finetune all our BERT models on MultiNLI using the Transformers library[2] using default parameters. The resulting BERT model is then frozen and used to produce features for our classifier. We use the following hyperparameters for our MLP classifier: number of feed-forward layers $\in$ [0,1], dimension of each layer $\in$ [64, 256], and dropout $\in$ [0.0, 0.3]. We optionally project the 768 dimensional BERT feature vector down to 64 dimensions. To train our classifier, we use the Adam optimizer (Kingma and Ba, 2014) with default parameters, and batch size $\in$ [32, 128].

We early stop our classifier based on the F1 score on the validation set, and choose the hyperparameters that obtain the best early-stopped F1 score on the validation set. For `Spouse` and `Disease`, we report the test F1 means and 95% confidence intervals of 5-10 runs. For `TACRED`, we follow Zhang et al. (2017), and report the test F1 of the median validation set F1 of 5 runs corresponding to the chosen hyperparameters.

A.2 Datasets

`Spouse` and `Disease` preprocessed datasets were obtained directly from the codebase provided by Hancock et al. (2018)[3]. We use the train, validation, test split provided by Hancock et al. (2018) for `Disease`, and split the development set of `Spouse` randomly into a validation and test set (the split was done at a document level). To process `TACRED`, we use the default BERT tokenizer and indexing pipeline in the Transformers library.

A.3 Explanations

The explanations can be found in Tables 6 and 7 on the following page. We use 40 explanations for `Spouse`, 28 explanations for `Disease`, and 128 explanations for `TACRED` (in accompanying file). The explanations were written by the authors.

[2] https://huggingface.co/transformers/

[3] https://worksheets.codalab.org/worksheets/0x900e7e41deaa4ec5b2fe41dc50594548/

$\{o_1\}$ and $\{o_2\}$ have a marriage license
$\{o_1\}$'s husband is $\{o_2\}$
$\{o_1\}$'s wife is $\{o_2\}$
$\{o_1\}$ and $\{o_2\}$ are married
$\{o_1\}$ and $\{o_2\}$ are going to tie the knot
$\{o_1\}$ married $\{o_2\}$
$\{o_1\}$ and $\{o_2\}$ are a married couple
$\{o_1\}$ and $\{o_2\}$ had a wedding
$\{o_1\}$ and $\{o_2\}$ married in the past
$\{o_1\}$ tied the knot with $\{o_2\}$
$\{o_1\}$ and $\{o_2\}$ have a son
$\{o_1\}$ and $\{o_2\}$ have a daughter
$\{o_1\}$ and $\{o_2\}$ have kids together
$\{o_1\}$ and $\{o_2\}$ are expecting a son
$\{o_1\}$ and $\{o_2\}$ are expecting a daughter
$\{o_1\}$ is engaged to $\{o_2\}$
$\{o_1\}$ is the fiancé of $\{o_2\}$
$\{o_1\}$ is the fiancée of $\{o_2\}$
$\{o_1\}$ is the daughter of $\{o_2\}$
$\{o_1\}$ is the mother of $\{o_2\}$
$\{o_1\}$ and $\{o_2\}$ are the same person
$\{o_1\}$ is the same person as $\{o_2\}$
$\{o_1\}$ is married to someone other than $\{o_2\}$
$\{o_1\}$ is the father of $\{o_2\}$
$\{o_1\}$ is the son of $\{o_2\}$
$\{o_1\}$ is marrying someone other than $\{o_2\}$
$\{o_1\}$ is the ex-wife of $\{o_2\}$
$\{o_1\}$ is a location
$\{o_2\}$ is a location
$\{o_1\}$ is an organization
$\{o_2\}$ is an organization
$\{o_1\}$ and $\{o_2\}$ are partners
$\{o_1\}$ and $\{o_2\}$ share a home
$\{o_1\}$ and $\{o_2\}$ are a couple
$\{o_1\}$ and $\{o_2\}$ share the same surname
someone is married to $\{o_1\}$
someone is married to $\{o_2\}$
$\{o_1\}$ is a person
$\{o_2\}$ is a person
$\{o_1\}$ and $\{o_2\}$ are different people

Table 6: Explanations for `Spouse`. The groups correspond to MARRIED, CHILDREN, ENGAGED, NEGATIVES and MISC.

The symptoms of $\{o_2\}$ appeared after the administration of $\{o_1\}$
$\{o_2\}$ developed after $\{o_1\}$
Patients developed $\{o_2\}$ after being treated with $\{o_1\}$
$\{o_1\}$ contributes indirectly to $\{o_2\}$
$\{o_1\}$ has been associated with the development of $\{o_2\}$
Symptoms of $\{o_2\}$ abated after withdrawal of $\{o_1\}$
A greater risk of $\{o_2\}$ was found in the $\{o_1\}$ group compared to a placebo
$\{o_2\}$ is a side effect of $\{o_1\}$
$\{o_2\}$ has been reported to occur with $\{o_1\}$
$\{o_2\}$ has been demonstrated after the administration of $\{o_1\}$
$\{o_1\}$ caused the appearance of $\{o_2\}$
Use of $\{o_1\}$ can lead to $\{o_2\}$
$\{o_1\}$ can augment $\{o_2\}$
$\{o_1\}$ can increase the risk of $\{o_2\}$
Symptoms of $\{o_2\}$ appeared after dosage of $\{o_1\}$
$\{o_1\}$ is a chemical
$\{o_2\}$ is a disease
$\{o_1\}$ is used for the treatment of $\{o_2\}$
$\{o_1\}$ is known to reduce the symptoms of $\{o_2\}$
$\{o_1\}$ is used for the prevention of $\{o_2\}$
$\{o_1\}$ ameliorates $\{o_2\}$
$\{o_1\}$ induces $\{o_2\}$
$\{o_1\}$ causes a disease other than $\{o_2\}$
$\{o_1\}$ is an organ
administering $\{o_1\}$ causes $\{o_2\}$ to worsen
$\{o_1\}$ is effective for the treatment of $\{o_2\}$
$\{o_1\}$ has an effect on $\{o_2\}$
$\{o_1\}$ has an attenuating effect on $\{o_2\}$

Table 7: Explanations for `Disease`