Few-Shot Aspect Extraction using Prompt Training

Daniel Korat[1,2], **Oren Pereg**[1], **Moshe Wasserblat**[1], **Kfir Bar**[2]
[1]Emergent AI Lab, Intel Labs, Israel
[2]The Efi Arazi School of Computer Science, Reichman University, Herzliya, Israel
[1]`firstname.lastname@intel.com`
[2]`firstname.lastname@post.idc.ac.il`

Abstract

A fundamental task of fine-grained sentiment analysis is aspect term extraction. Supervised-learning approaches have demonstrated state-of-the art results for this task; however, they underperform in few-shot scenarios, where labeled training data is scarce. Prompt-based training has proven effective in few-shot sequence classification; however, it would not apply to token classification tasks. In this work we propose PATE (**P**rompt-based **A**spect **T**erm **E**xtraction), a few-shot prompt-based method for the token classification task of aspect term extraction. We demonstrate that this method significantly outperforms the standard supervised training approach in few-shot setups and make our code publicly available[1].

1 Introduction

A fundamental task of fine-grained sentiment analysis is aspect term extraction (ATE). ATE aims to detect aspects that may have associated opinions in a textual review. For example, in the sentence *"The battery life is amazing"*, the aspect term is *battery life* (and the associated opinion term is *amazing*). ATE is typically formulated as a supervised token classification task. RNN-based models (Liu et al., 2015) and Transformer-based models (Vaswani et al., 2017) showed promising results when trained on thousands of labeled sentences. However, real-world deployments are often required to operate in few-shot environments where labeled training data is scarce and costly.

Recent few-shot prompt-based methods such as PET (Schick and Schütze, 2021a,b) and ADAPET (Tam et al., 2021) achieve state-of-the-art results in data-scarce environments. They convert the classification task into a masked language modeling (MLM) objective, which pre-trained language models are typically trained on, and excel in. Those prompt-based models output tokens in a cloze-style format that map to the corresponding labels via a predefined template. However, those methods are designed for sequence classification tasks and would not apply for token classification tasks. In this paper we propose PATE, a prompt-based method for token classification that is applied for the ATE task. PATE works by first extracting token-level aspect candidates and then using prompts for generating cloze-style yes/no questions for filtering out non-aspect candidates.

The contribution of this paper is twofold. First, we propose PATE, a method for using prompt-based training for aspect term extraction. Second, we show that this method significantly outperforms the standard pre-trained language model (PLM) fine-tuning approach in few-shot scenarios.

2 Related Work

The most active line of ATE related work is based on token-level supervised learning (Liu et al., 2015; Xu et al., 2018, 2019; Karimi et al., 2021). Although they achieve promising results, these methods require large amounts of labeled data, especially when training large transformer models.

[1]`https://github.com/IntelLabs/aspect-extraction`

There are several lines of work that address labeled data scarcity in ATE. One approach employs domain adaptation techniques, namely, utilizing existing labeled data from one domain to adapt a model to another domain. Ding et al. (2017) proposed using domain-invariant dependency-based aspect extraction rules as auxiliary supervision for an RNN model. However, this method depends on the quality of manually-crafted rules. Wang and Pan (2019) addressed this issue by designing a dependency prediction task that encodes dependency relations into the hidden representations of words. Pereg et al. (2020) proposed to utilize external dependencies, injected into the self-attention mechanism during fine-tuning of pre-trained Transformers. Lekhtman et al. (2021) introduced a pivot-based method for pre-training contextual word embeddings. Another line of work addresses zero-shot ATE. Recently, Shu et al. (2022) demonstrated that ATE can be successfully presented as a natural language inference task, achieving state-of-the-art results for zero-shot. However, the accuracy degradation compared to supervised methods is significant.

Recently, prompt-based few-shot methods have demonstrated impressive results in sequence classification tasks. A well known example is Pattern Exploiting Training (PET) (Schick and Schütze, 2021a,b) which leverages patterns for few-shot learning, by reformulating natural language tasks as cloze-style questions. ADAPET (Tam et al., 2021) improves PET by modifying its objective to provide denser supervision during fine-tuning, alleviating the need for task-specific unlabeled data.

As opposed to other ATE methods, our approach does not rely on specialized neural architecture or handcrafted rules. Instead, we build upon the inherent token prediction capability of PLMs. Our work employs a preliminary step of detecting potential aspects within the text, and a cloze pattern generation step for filtering out non-aspect candidates.

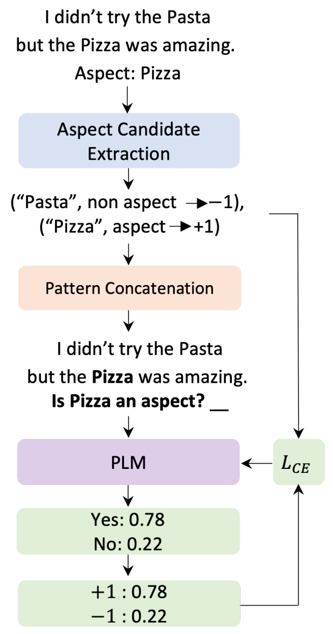

Figure 1: PATE training: First, aspect candidates are extracted and are associated with labels according to the training set, then, yes/no cloze questions are concatenated and a PLM is trained to minimize the cross entropy loss (L_{CE}) from the correct answer.

3 PATE

Figure 1 is an illustration of PATE training process. First, aspect candidates are extracted from the input sentence. The candidates are associated with labels according to the training set. Next, yes/no cloze questions and their answers, aimed to filter out non-aspect candidates, are generated. Each cloze question is then concatenated to the input instance and finally, a PLM is fine-tuned to minimize the loss between the predicted answer and the correct answer.

Aspect Candidate Extraction. Cloze question patterns were shown to be effective in few-shot sequence classification (Schick and Schütze, 2021a; Tam et al., 2021; Zheng et al., 2022). However, the goal in token classification is more fine-grained. Our approach for exploiting cloze questions in ATE is based on introducing a pre-process mechanism that extracts aspect candidates. The aspect candidates are represented as text spans, that are of higher probability to function as aspects based on the context in which they appear. For this purpose we implement two aspect candidate extraction (ACE) methods. The first ACE method follows previous work that use noun and noun phrase detection for aspect candidate extraction (Hu and Liu, 2004b; Tulkens and van Cranenburgh, 2020). Along this line, we use a simple rule-based noun phrase extractor, based on patterns of part-of-speech tags (Subhashini and Kumar, 2010; Chakraborty et al., 2016) for extracting aspect candidates. Formally, the output is denoted by $T^C = \{x_i, asp_i^{cand}\}$ where x_i is a sentence and asp_i^{cand} is a noun phrase that constitutes an aspect candidate in the context of x_i.

In order to complement the first ACE method with non-noun aspect candidates, we introduce a second ACE method. This method, based on a neural network, employs a PLM that is fine-tuned using the available labeled data for the ATE task, in a similar fashion to the baseline implementation (see Section 4). Then, this PLM is used to extract aspect candidates, generating $T^{neu} = \{x_i, asp_i^{neu}\}$. At

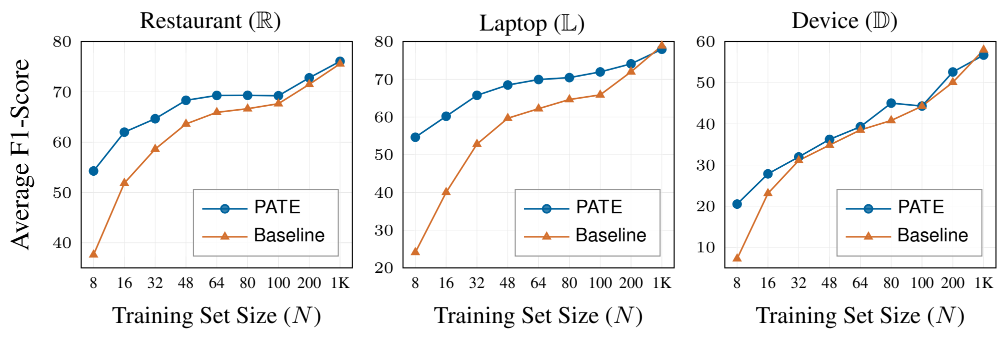

Figure 2: F1 as a function of training set size N of PATE and the baseline model on $\mathbb{R}$, $\mathbb{L}$ and $\mathbb{D}$

inference time, the final output of the ACE step is produced by unifying both ACE methods, denoting $T^C = T^C \cup T^{neu} = \{x_i, asp_i^{cand}\}$.

Training Set Generation. Given a small gold set of labeled examples $T^G = \{x_i, asp_i^g\}$ where x_i is a sentence and asp_i^g is a gold aspect in the context of x_i, and a set of examples containing aspect candidates T^C, we generate a training set T by unifying the examples from T^G and T^C such that $T = T^G \cup T^C = \{x_i, asp_i, y_i\}$ where x_i is a sentence, asp_i is a span of text within x_i, and $y_i \in (0, 1)$ indicates whether asp_i is an aspect ($y_i = +1$) or a non aspect ($y_i = -1$). We set y_i to be $+1$ for all the examples in T^G and set y_i to be -1 for all the examples that are not in the gold training set. Formally:

$$y_i = \begin{cases} +1 & asp_i \in T^G \\ -1 & asp_i \notin T^G \end{cases}$$

Pattern Generation for Aspect Candidate Filtering. Given an input sentence x and a span of text asp in x, our goal is to predict whether asp is an aspect ($y = +1$) or not ($y = -1$) in the context of x. For this purpose we devise a pattern-based objective via a function P that inputs a sentence x and a span of text asp, and outputs a sentence $P(x, asp)$ that contains a yes/no question and exactly one mask token. We then define a verbalizer v that maps between the label y of (x, asp) and a word in the PLM vocabulary, namely we set v = "Yes" if ($y = +1$) which indicates that asp is indeed an aspect, and set v = "No" if ($y = -1$) which indicates non aspect. For example, given the following input pair: (x, asp) = (The dessert was amazing, *dessert*), The task is redefined as concatenating to x a question asking whether the most likely choice in the masked position regarding asp (dessert) is "Yes" or "No", denoting: $P(x, asp)$ = The dessert was amazing. Does the review focus on *dessert*? __

Training and Inference. For each triplet (x, asp, y) in training set T, we generate a pair (p, v) where $p = P(x, asp)$ and v is the label verbalizer. We adapt the score for label y given input x generated by a PLM for a masked token defined by Schick and Schütze (2021a,b) to incorporate asp, denoting:

$$S_{P,v}(y|(x, asp)) = PLM(v(y)|P(x, asp)) \tag{1}$$

We then use the cross entropy between the softmaxed probability distribution of $S_p(y|(x, asp))$ and the true distribution of the training example summed over all the training examples $\{x_i, asp_i, y_i\}$ in T to fine-tune the PLM. Similarly, at inference time, Equation 1 is used to classify candidates as aspects/non-aspects by calculating their yes/no verbalizer scores. Following studies that show benefits of continued pretraining (CPT) of PLMs in general (Howard and Ruder, 2018; Gururangan et al., 2020) and in few-shot setups (Schick and Schütze, 2021a), we use unlabeled examples from the domain of the labeled data to train the PLM with an MLM objective, prior to fine-tuning.

4 Data and Experiments

Following previous ATE work, we conduct experiments on datasets of customer reviews from three different domains: Restaurant ($\mathbb{R}$), Laptop ($\mathbb{L}$) and Device ($\mathbb{D}$). $\mathbb{R}$ includes restaurant reviews from

SemEval 2014 (Pontiki et al., 2014) and SemEval 2015 (Pontiki et al., 2015). $\mathbb{L}$ includes laptop reviews from SemEval 2014, and $\mathbb{D}$ is provided by Hu and Liu (2004a) and contains reviews of five different digital products.

Systematically evaluating few-shot performance can be challenging, as fine-tuning using small datasets may incur instability (Dodge et al., 2020; Zhang et al., 2021), and results may change dramatically given different random data selections. Thus, we adopt a rigorous framework for training and evaluating few-shot methods, proposed by Zheng et al. (2022). Each dataset is first split into D_{test} (1/3) and D_{train} (2/3).

To tune the model for N labeled training examples, we select N examples from D_{train}, denoted D_N. Then, we apply a multi-split strategy, wherein D_N is randomly divided into equally sized D^k_{train} and D^k_{dev}. This division process is repeated $K = 5$ times.

N	P_0	P_1	P_2	P_3
8	**54.6** ± 2.7	53.3 ± 6.4	53.6 ± 2.6	51.3 ± 6.2
16	60.2 ± 1.1	62.2 ± 0.8	**62.8** ± 1.4	61.4 ± 0.5
32	**65.8** ± 1.1	64.4 ± 0.9	64.4 ± 0.9	62.7 ± 1.3
64	69.9 ± 1.2	68.7 ± 0.5	69.4 ± 1.1	**70.0** ± 0.5
100	**72.0** ± 0.3	70.2 ± 1.3	70.8 ± 1.2	70.7 ± 1.9
200	**74.1** ± 0.7	73.2 ± 0.9	**74.1** ± 0.5	**74.1** ± 1.1
1000	78.0 ± 0.5	**78.2** ± 0.5	78.0 ± 0.4	**78.2** ± 0.8

Table 1: Average F1 and standard deviation for PATE on Laptops dataset using different cloze-patterns, for various training set sizes N

Given a hyper-parameter space H, for each $h \in H$ and $k \in 1...5$, we train PATE on D^k_{train} using h and evaluate it on D^k_{dev}. Let h^* be the hyper-parameter set that achieves the best mean F1-score across all k splits. To test the model performance, we train it using h^* on D_N, and evaluate on D_{test}. Finally, we report the mean F1-score for this test over 3 random seeds. The selected h^* per dataset appear in Table 6 in Appendix A.2. CPT always uses the full D_{train} (unlabeled), containing **2,565**, **3,896** and **2,557** examples for $\mathbb{L}$, $\mathbb{R}$ and $\mathbb{D}$, respectively. We adopt the HuggingFace implementation[2] of RoBERTa-base (Liu et al., 2019) for both PATE and the baseline.

Baseline Previous work in low-resource ATE, including those mentioned in Section 2, are applicable to domain-adaption and zero-shot setups; however, they do not apply to few-shot scenarios like ours. Hence, our standard supervised training baseline is based on the common approach of fine-tuning a PLM for ATE. Specifically, we formulate the ATE task as a token classification task (Poria et al., 2016; Xu et al., 2018) and fine-tune the RoBERTa-base model with a token classification layer using the few-shot labeled data. For fair comparison, we tune the baseline hyper-parameters in the same manner performed for PATE.

5 Results

Figure 2 shows a comparison between PATE and the baseline for different values of N; PATE significantly outperforms the baseline in small N values. For example, for $N = 8$, PATE outperforms the baseline by **16.6**, **30.5**, and **13.3** average F1 on $\mathbb{R}$, $\mathbb{L}$ and $\mathbb{D}$, respectively. As N increases, the performance gains decrease; however, PATE outperforms the baseline for $N \leq 1000$ in $\mathbb{R}$ and for $N \leq 200$ in $\mathbb{L}$. Detailed results are shown in Tables 3-5 in Appendix A.1. These results highlight the model's ability to better leverage the inherent knowledge of the PLM, by using cloze-style patterns during training.

We evaluated PATE using four different cloze patterns. We chose simple and intuitive patterns which requires no expertise in prompt selection. Namely, we ask a Yes/No question about whether some candidate in the sentence is an aspect or not (see details in Appendix A.3). The results in Table 1 show that pattern selection has a very small effect on the performance and there was no single prominent pattern.

N	**Method**	$\mathbb{L}$	$\mathbb{R}$
16	Baseline	40.0 ± 7.3	51.9 ± 0.9
	PATE	**60.2** ± 1.1	**62.0** ± 1.2
	– n-ACE	56.1 ± 3.7	55.1 ± 5.2
	– CPT	51.7 ± 4.4	56.5 ± 4.7
64	Baseline	62.2 ± 1.6	65.9 ± 1.1
	PATE	**69.9** ± 1.1	**69.3** ± 1.6
	– n-ACE	64.2 ± 0.9	64.2 ± 1.3
	– CPT	65.5 ± 3.6	65.6 ± 1.1
128	Baseline	68.0 ± 2.2	69.6 ± 0.4
	PATE	**72.4** ± 0.4	**69.7** ± 1.9
	– n-ACE	65.3 ± 0.1	66.7 ± 0.3
	– CPT	71.5 ± 1.1	68.4 ± 1.7

Table 2: PATE ablation test showing average F1 and standard deviation for Laptop and Restaurant datasets on three training set sizes N, and when excluding n-ACE or CPT.

[2]`https://github.com/huggingface/transformers`

In fact, using the null-pattern P_3, composed only of the candidate aspect followed by a masked Yes/No token, yielded F1 comparable to other patterns. This demonstrates the robustness of PATE for pattern selection.

To study the effect of the neural ACE and CPT on the model's performance, we conducted ablation experiments across three values of N. The results, in Table 2, show that both steps hold a significant contribution, especially for $N \leq 64$. CPT is most valuable at $N = 16$, and its contribution decreases as N increases. The gains from neural ACE are significant across all tested values for N.

6 Conclusion

We propose a method for using prompt-based training in the form of cloze questions for few-shot token classification applied for aspect term extraction. We demonstrate that this method outperforms the standard supervised training baseline in few-shot setups and is robust to pattern selection. This method better handles real-world scenarios where labeled data is scarce.

References

Neelotpal Chakraborty, Samir Malakar, Ram Sarkar, and Mita Nasipuri. 2016. A rule based approach for noun phrase extraction from english text document. In *Seventh International Conference on CNC*, pages 13–26.

Ying Ding, Jianfei Yu, and Jing Jiang. 2017. Recurrent neural networks with auxiliary labels for crossdomain opinion target extraction. In *Association for the Advancement of Artificial Intelligence*, pages 3436—3442.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. 2020. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv preprint arXiv:2002.06305*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Minqing Hu and Bing Liu. 2004b. Mining opinion features in customer reviews. In *American Association for Artificial Intelligence*.

Akbar Karimi, Leonardo Rossi, and Andrea Prati. 2021. Improving BERT performance for aspect-based sentiment analysis. In *Proceedings of The Fourth International Conference on Natural Language and Speech Processing (ICNLSP 2021)*, pages 39–46, Trento, Italy. Association for Computational Linguistics.

Entony Lekhtman, Yftah Ziser, and Roi Reichart. 2021. DILBERT: Customized pre-training for domain adaptation with category shift, with an application to aspect extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 219–230, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1433–1443, Lisbon, Portugal. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. Cite arxiv:1907.11692.

Oren Pereg, Daniel Korat, and Moshe Wasserblat. 2020. Syntactically aware cross-domain aspect and opinion terms extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1772–1777, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 486–495, Denver, Colorado. Association for Computational Linguistics.

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. SemEval-2014 task 4: Aspect based sentiment analysis. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 27–35, Dublin, Ireland. Association for Computational Linguistics.

Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2016. Aspect extraction for opinion mining with a deep convolutional neural network. *Knowledge-Based Systems*, 108:42–49. New Avenues in Knowledge Bases for Natural Language Processing.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, Online. Association for Computational Linguistics.

Timo Schick and Hinrich Schütze. 2021b. Generating datasets with pretrained language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6943–6951, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Lei Shu, Jiahua Chen, Bing Liu, and Hu Xu. 2022. Zero-shot aspect-based sentiment analysis. *arXiv preprint arXiv:2202.01924*.

R Subhashini and V Jawahar Senthil Kumar. 2010. Shallow nlp techniques for noun phrase extraction. In *Trendz in Information Sciences & Computing (TISC2010)*, pages 73–77. IEEE.

Derek Tam, Rakesh R. Menon, Mohit Bansal, Shashank Srivastava, and Colin Raffel. 2021. Improving and simplifying pattern exploiting training. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4980–4991, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Stéphan Tulkens and Andreas van Cranenburgh. 2020. Embarrassingly simple unsupervised aspect extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3182–3187, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Wenya Wang and Sinno Jialin Pan. 2019. Syntactically meaningful and transferable recursive neural networks for aspect and opinion extraction. *Computational Linguistics*, 45(4):705–736.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. 2019. BERT post-training for review reading comprehension and aspect-based sentiment analysis. 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 2324–2335, Minneapolis, Minnesota. Association for Computational Linguistics.

Hu Xu, Bing Liu, Lei Shu, and Philip S. Yu. 2018. Double embeddings and CNN-based sequence labeling for aspect extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 592–598, Melbourne, Australia. Association for Computational Linguistics.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2021. Revisiting few-sample {bert} fine-tuning. In *International Conference on Learning Representations*.

Yanan Zheng, Jing Zhou, Yujie Qian, Ming Ding, Chonghua Liao, Li Jian, Ruslan Salakhutdinov, Jie Tang, Sebastian Ruder, and Zhilin Yang. 2022. FewNLU: Benchmarking state-of-the-art methods for few-shot natural language understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 501–516, Dublin, Ireland. Association for Computational Linguistics.

A Appendix

A.1 Full results

We provide average precision, recall and F1 scores (and their standard deviations), per dataset, in Tables 3-5. The overall relative advantage in recall can be attributed in part to the ACE step, due to its two different strategies for discovering aspect candidates (POS features and token classification) which may complement each other.

N	**Method**	**P**	**R**	**F1**
8	Baseline	53.1 ± 8.8	31.1 ± 9.3	37.6 ± 7.1
8	PATE	53.6 ± 5.9	56.7 ± 8.1	54.3 ± 0.3
16	Baseline	47.6 ± 1.6	57.2 ± 4.0	51.9 ± 0.9
16	PATE	54.7 ± 3.9	72.2 ± 5.0	62.0 ± 1.2
32	Baseline	57.4 ± 5.3	61.5 ± 9.3	58.6 ± 3.8
32	PATE	61.1 ± 5.7	70.2 ± 8.7	64.7 ± 2.3
48	Baseline	59.5 ± 4.2	68.6 ± 1.0	63.6 ± 2.1
48	PATE	64.2 ± 2.0	73.0 ± 2.8	68.3 ± 1.9
64	Baseline	62.0 ± 1.4	70.4 ± 2.0	65.9 ± 1.1
64	PATE	67.1 ± 1.5	71.7 ± 1.4	69.3 ± 0.2
80	Baseline	64.7 ± 2.7	69.0 ± 3.1	66.7 ± 0.5
80	PATE	68.1 ± 0.7	70.7 ± 3.1	69.3 ± 1.5
100	Baseline	63.6 ± 2.2	72.4 ± 1.3	67.7 ± 0.7
100	PATE	67.0 ± 1.9	71.7 ± 1.1	69.2 ± 1.4
200	Baseline	68.8 ± 0.7	74.4 ± 0.4	71.5 ± 0.5
200	PATE	70.7 ± 1.5	75.1 ± 1.6	72.8 ± 0.8
1000	Baseline	74.7 ± 1.0	76.5 ± 0.8	75.6 ± 0.2
1000	PATE	74.4 ± 0.5	77.8 ± 0.3	76.1 ± 0.3

Table 3: Average Precision (P), Recall (R), F1 and standard deviation for the baseline and PATE on Restaurant dataset for various training set sizes N.

A.2 Hyper-parameters

Hyper-parameter tuning was performed as described in Section 4. For the baseline, we try values in the range $[1\text{e}{-5}, 3\text{e}{-5}]$ for the `learning_rate`, $[400, 2000]$ for the number of training steps (`max_steps`) and $[8, 16]$ for the batch size. For PATE, the search values are:
`max_steps` $\in \{700, 1000\}$,
`learning_rate` $\in \{2\text{e}{-5}, 3\text{e}{-5}\}$,
`neu_ace_max_steps` $\in \{500, 1000\}$,
`neu_ace_learning_rate` $\in \{2\text{e}{-5}, 3\text{e}{-5}\}$,
`CPT_max_steps` $\in \{1000, 2000\}$,

N	Method	P	R	F1
8	Baseline	34.7 ± 6.4	19.5 ± 5.2	24.1 ± 3.5
8	PATE	55.9 ± 9.7	55.7 ± 8.0	54.6 ± 2.7
16	Baseline	38.3 ± 8.4	42.6 ± 7.5	40.0 ± 7.3
16	PATE	59.4 ± 5.5	62.5 ± 7.8	60.2 ± 1.1
32	Baseline	51.1 ± 0.6	54.8 ± 3.0	52.8 ± 1.7
32	PATE	63.4 ± 2.3	68.4 ± 1.8	65.8 ± 1.1
48	Baseline	54.2 ± 0.9	64.0 ± 1.9	58.7 ± 1.0
48	PATE	65.8 ± 1.5	71.6 ± 3.5	68.5 ± 1.5
64	Baseline	60.4 ± 2.5	64.2 ± 1.5	62.2 ± 1.6
64	PATE	67.7 ± 3.7	72.7 ± 3.6	69.9 ± 1.2
80	Baseline	64.0 ± 0.6	65.4 ± 2.9	64.6 ± 1.3
80	PATE	68.7 ± 2.1	72.4 ± 3.1	70.4 ± 0.9
100	Baseline	64.8 ± 1.8	67.3 ± 3.7	65.9 ± 1.0
100	PATE	70.0 ± 1.5	74.1 ± 1.6	72.0 ± 0.3
200	Baseline	71.2 ± 0.6	72.8 ± 2.2	72.0 ± 1.4
200	PATE	72.8 ± 1.2	75.4 ± 0.2	74.1 ± 0.7
1000	Baseline	76.6 ± 0.6	81.3 ± 0.3	78.9 ± 0.2
1000	PATE	76.0 ± 0.6	80.1 ± 1.6	78.0 ± 0.5

Table 4: Average Precision (P), Recall (R), F1 and standard deviation for the baseline and PATE on Laptop dataset for various training set sizes N.

N	Method	P	R	F1
8	Baseline	26.8 ± 24.8	4.2 ± 3.8	7.2 ± 6.7
8	PATE	31.7 ± 6.8	16.2 ± 4.5	20.5 ± 3.4
16	Baseline	38.6 ± 11.1	17.3 ± 6.3	23.1 ± 6.1
16	PATE	38.0 ± 11.3	23.1 ± 2.7	27.9 ± 4.2
32	Baseline	35.1 ± 8.5	28.8 ± 10.4	31.1 ± 8.8
32	PATE	41.4 ± 4.5	26.8 ± 8.3	32.0 ± 7.2
48	Baseline	41.1 ± 8.7	30.7 ± 3.9	34.9 ± 5.0
48	PATE	41.6 ± 2.5	32.4 ± 6.7	36.2 ± 5.1
64	Baseline	45.9 ± 3.2	33.3 ± 0.5	38.5 ± 0.9
64	PATE	45.3 ± 0.1	35.0 ± 4.8	39.3 ± 3.1
80	Baseline	47.0 ± 3.0	36.8 ± 6.1	40.8 ± 3.0
80	PATE	50.7 ± 1.4	40.7 ± 4.5	45.0 ± 3.3
100	Baseline	47.9 ± 1.4	41.5 ± 4.2	44.3 ± 1.7
100	PATE	48.7 ± 2.8	40.8 ± 1.7	44.3 ± 0.9
200	Baseline	55.5 ± 5.1	45.7 ± 3.2	50.1 ± 3.8
200	PATE	52.2 ± 4.5	53.3 ± 1.6	52.6 ± 1.6
1000	Baseline	56.0 ± 1.2	60.2 ± 1.6	58.0 ± 1.3
1000	PATE	53.5 ± 1.7	60.4 ± 1.8	56.7 ± 0.6

Table 5: Average Precision (P), Recall (R), F1 and standard deviation for the baseline and PATE on Device dataset for various training set sizes N.

`CPT_learning_rate`$\in\{1e{-}5, 2e{-}5, 3e{-}5\}$. We provide the selected parameters for each setup, in Table 6. Note that the batch size is always 8 where not specified.

A.3 Patterns

We tested cloze patterns with varying content words, word order and length. Table 1 shows results for P_0 – the pattern used for the primary tests – as well as 3 additional patterns P_1, P_2, P_3. P_3 is a null-pattern, that is, it does not contain any tokens other than the candidate aspect followed by a

masked Yes/No token. Given an input sentence x_i and a span of text asp_i in x_i, these are the patterns:
$P_0(x, asp^c) = x$. So, does the review in the previous sentence focus on asp^c? __
$P_1(x, asp^c) = x$. Is asp^c an aspect? __
$P_2(x, asp^c) = x$. So, is the review about asp^c? __
$P_3(x, asp^c) = x\ asp^c$ __

	Baseline			PATE		
Parameter	$\mathbb{L}$	$\mathbb{R}$	$\mathbb{D}$	$\mathbb{L}$	$\mathbb{R}$	$\mathbb{D}$
`adam_epsilon`	1e-8	1e-8	1e-8	1e-8	1e-8	1e-8
`max_seq_length`	128	128	128	128	128	128
`mlm_probability`	–	–	–	0.15	0.15	0.15
`learning_rate`	3e-5	3e-5	3e-5	2e-5	3e-5	2e-5
`per_device_train_batch_size`	16	16	16	8	8	8
`max_steps`	600	800	1000	700	1000	1000
`neural_ace_max_steps`	–	–	–	500	1000	500
`neural_ace_learning_rate`	–	–	–	2e-5	2e-5	2e-5
`CPT_max_steps`	–	–	–	1000	2000	2000
`CPT_learning_rate`	–	–	–	2e-5	3e-5	2e-5

Table 6: Hyperparameters for baseline and PATE per dataset