Generating unlabelled data for a tri-training approach in a low resourced NER task

Hugo Boulanger, Thomas Lavergne, Sophie Rosset
Université Paris-Saclay, CNRS,
Laboratoire Interdisciplinaire des Sciences du Numérique, 91400, Orsay, France
first-name.last-name@lisn.upsaclay.fr

Abstract

Training a tagger for Named Entity Recognition (NER) requires a substantial amount of labeled data in the task domain. Manual labeling is a tedious and complicated task. Semi-supervised learning methods can reduce the quantity of labeled data necessary to train a model. However, these methods require large quantities of unlabeled data, which remains an issue in many cases.

We address this problem by generating unlabeled data. Large language models have proven to be powerful tools for text generation. We use their generative capacity to produce new sentences and variations of the sentences of our available data. This generation method, combined with a semi-supervised method, is evaluated on CoNLL and I2B2. We prepare both of these corpora to simulate a low resource setting. We obtain significant improvements for semi-supervised learning with synthetic data against supervised learning on natural data.

1 Introduction

Training models to solve NER tasks requires a considerable amount of labeled data. In most NLP tasks, this data needs to be related to the task domain and must be in the targeted language. While English is a well-covered language, corpora are still being built to cover new domains or expand existing ones. For any other languages, corpora cover fewer domains. Data in the private sector is rarely shareable due to privacy reasons. It is also the case in domains such as the medical domain.

Recent approaches tackle the issue of the absence of resources by leveraging knowledge or data from other sources. Zero-shot learning is a learning paradigm trying to solve a target task without any labeled data. It uses the knowledge of how to predict labels of an adjacent task and applies it to predict the unseen labels of the target task (Wang et al., 2019). We do not aim to solve the NER problem in a situation with such strict data restrictions. Labeling a few examples is almost always possible. Few-shot learning provides training methods to generalize from a few labeled examples. These methods use the labeled examples to build representations of the class, which serve as comparison points for inference (Dopierre et al., 2021). Transfer learning leverages the knowledge learned on tasks of the domain to improve the performance on a specific task (Ruder, 2019). It is quite common to see cross-lingual transfer from higher-resourced languages where the task exists. However, the most prominent use case of transfer learning in NLP is the use of language models for data representation. We use this type of transfer learning to build high-performing taggers from BERT models. Semi-supervised learning is a paradigm where unlabeled data is widely available. The unlabeled data is used to improve the model's performance by giving a better topology of the data space.

We propose to use a semi-supervised learning method in a context where data is scarce enough to be fully labeled. We aim to achieve this by using large language models to generate the necessary unlabeled data. We test whether large language models can generate data that make tri-training a viable option in a low-resource context. The performances of our baseline models are compared against the performances of the ensembles of models trained with tri-training on CoNLL (Sang and De Meulder, 2003) and I2B2 (Uzuner et al., 2011). Significant improvements are observed using our method on the reduced datasets.

Language modeling has already been used as an augmentation method to generate labeled and unlabeled examples for NER in DAGA (Ding et al., 2020). However, our taggers overperform the taggers presented on the gold standard by 30 points at size 1000 and 9 points at full size. The semi-supervised method used in DAGA, self-training, is also prone to errors due to reinforcement of early mistakes. In our case, we generate unlabeled sen-

Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing, pages 30 - 37

tences using pre-trained large language models. We test this method with subsets of data ranging from 50 examples to 1000 examples vs. over 1000 in DAGA.

Thus, our main contribution is using out-of-the-box large language models as tools to obtain unlabeled data for semi-supervised learning in NER in a low-resource setting. The code relative to the experiment will be available in a public repository[1].

Section 2 presents state of the art related to data augmentation, semi-supervised learning in NER, and language modeling. Section 3 presents tri-training (Zhou and Li, 2005), and how we fit generation into it. Section 4 touches on the technical details of the experiments. Section 5 and 6 are the discussion and the conclusion of the article.

2 Related Works

Learning models in a low-resource setting require extracting every possible information from the available data. Data augmentation is a common technique that creates synthetic data from available data. In Natural Language Processing, augmentation is used across various tasks to help achieve better performances. In classification, techniques such as back-translation (Sennrich et al., 2016) or Easy Data Augmentation (Wei and Zou, 2019) are used. However, in tagging, paraphrasing using back-translation (Neuraz et al., 2018) is not bringing significant improvements. Recent works show that using language models learned on the training data to generate labeled and unlabeled examples can bring improvements (Ding et al., 2020).

Inductive semi-supervised learning (Van Engelen and Hoos, 2020) aims at improving the performances of models through the addition of unlabeled data. For Named Entity Recognition, *pseudo-labeling* is a method that has been used (Chen et al., 2019). *Pseudo-labeling* is one of the semi-supervised learning methods. The unlabeled data receives pseudo-labels from the models trained. This pseudo-labeled data is then used alongside labeled data to train the models. Variants of the method exists (Yarowsky, 1995) (McClosky et al., 2006) (Blum and Mitchell, 1998) with varying quantities of models trained. The separation of the data between the different models trained and how the models are used to produce pseudo-labels also creates variants to this method. In our case, we use tri-training (Zhou and Li, 2005), which uses three models. This method has been used to solve Clinical Concept Extraction in the medical domain (Chen et al., 2019) on new data.

Semi-supervised learning methods still require a significant amount of unlabeled data. However, with current advances in language modeling, this method could be improved. Transformer-based models (Vaswani et al., 2017) have been a revolution in the language modeling landscape. From their first iterations like GPT (Radford et al., 2018) to their more recent ones like T5 (Raffel et al., 2020) and GPT-3 (Brown et al., 2020), transformer-based models have become a staple of Natural Language Processing as fine-tuning or transferring knowledge from these models often outperforms learning a model on the task directly. While our taggers are based on BERT models (Devlin et al., 2018), we otherwise use the generative power of GPT2 (Radford et al., 2019) to provide unlabeled data for the semi-supervised training. GPT2 has been finetuned and used to generate unlabeled data for classification in a high resource context (He et al., 2021).

3 Methods

This section provides details on the tri-training process for sentence tagging and how we levy language modeling as an unlabeled data provider.

3.1 Tri-training

Algorithm 1 Tri-training ((Zhou and Li, 2005), (Ruder and Plank, 2018))

for $i \in \{1..3\}$ **do**
  $m_i \leftarrow train\_model(sampling(L), m_i)$
while Any m_i still learns **do**
  for $i \in \{1..3\}$ **do**
    $L_i \leftarrow \emptyset$
    $j, k \leftarrow \{1..3\} - |i|$
    for $x \in U$ **do**
      if $m_j(x) = m_k(x)$ **then**
        $L_i \leftarrow L_i \cup \{(x, m_j(x))\}$
  for $i \in \{1..3\}$ **do**
    $m_i \leftarrow train\_model(L_i \cup L, m_i)$

Tri-training is an inductive semi-supervised learning (Van Engelen and Hoos, 2020) method using an ensemble of three models. The models are trained in a supervised learning manner on a set of labeled and pseudo-labeled data. As we try to solve

[1] https://github.com/HugoBoulanger/Tritraining-Gen

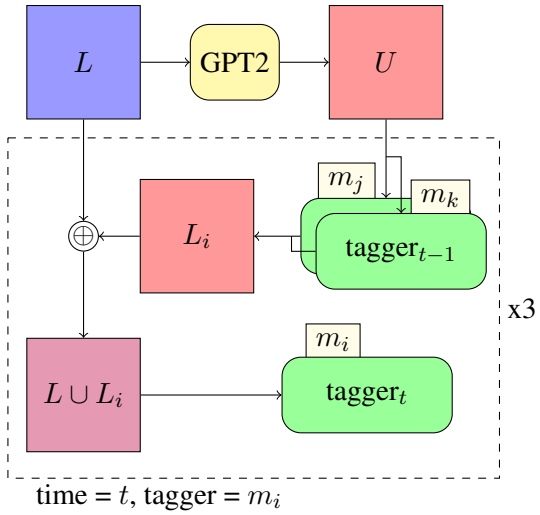

Figure 1: Tri-training with unlabeled data U generation. In rectangles are the data sets, and in rounded rectangles are the different models. The procedure is shown at episode t for model m_i. The initialization is not represented and is done by sampling with replacement from L.

a NER task, the models we use for the ensemble are taggers. Further description of the taggers can be found in the experiments section. We describe the Algorithm 1 in the following paragraphs, and we show our additions in Figure 1.

Tri-training. Tri-training is an episodic training method that stops when each model of the ensemble has stopped improving. The most crucial feature of tri-training is the construction of the training set of the models. This is shown from line 4 to line 9 in Algorithm 1 and in the second line of Figure 1. For each model m_i, a pseudo-labeled set L_i is constructed. L_i is composed of the unlabeled sentences $x \in U$ for which the predictions of the models m_j and m_k $i \notin \{j, k\}$ are equal. These predictions are added to L_i alongside x as their pseudo-labels. A threshold can also be used to remove uncertain annotations. However, it was concluded that it was not necessary for simple tri-training (Ruder and Plank, 2018). The models are then trained on both the natural and synthetic data $L \cup L_i$. L is the labeled training data. In our case, it represents any subset of the training corpus made for the low resource setting as explained in section 4.2. The operations described above are repeated until all models have stopped learning.

Initialization. The central part of Algorithm 1 described above assumes that models are sufficiently trained and different to create varied pseudo-labels. To achieve these prerequisites, we pre-train the models. The models m_i are pre-trained on different random subsets of the labeled data L. These subsets are made by sampling with replacement from the training set. This operation is also referred to as *bootstrap sampling* in (Zhou and Li, 2005). Sampling the pre-training data is done to introduce variety in the train sets of the three models without incurring performance losses.

Inference. For inference, we obtain an ensemble of 3 different models that can be used together with a voting system. We keep the labels with the highest summed score across the three models.

As a semi-supervised learning algorithm, tri-training requires a substantial amount of unlabeled examples. The specificity of our study is the use of a generator to create the unlabeled examples.

3.2 Generation

Applying semi-supervised learning methods is more complicated when there is no unlabeled data. We used the text of the labeled data as the context for the generation model. We use the generation model in two different ways: (i) follow-up sentence generation and (ii) sentence completion, as shown in Figure 2.

The first generation method we use is follow-up sentence generation. Large language models like GPT-2 (Radford et al., 2019) are trained on texts containing multiple sentences. This kind of model should be able to generate the follow-up sentence from the context. Using these models out-of-the-box should work without any finetuning. We apply follow-up sentence generation to generate new examples. With this method, we aim to generate new sentences that are within the same domain but have different structures.

The second method we use is sentence completion. We remove the end of the sentence and complete it using the language model for this method. We aim to generate alternative contexts to the part of the sentence we keep with this method. While this method might bring more variations by taking out random portions of the sentences, it is easier to use this way.

3.3 Evaluation

We aim at evaluating whether the data generated with large language models is of sufficient quality to serve as unlabeled data in a tri-training scenario. To that end, we evaluate the performances of the

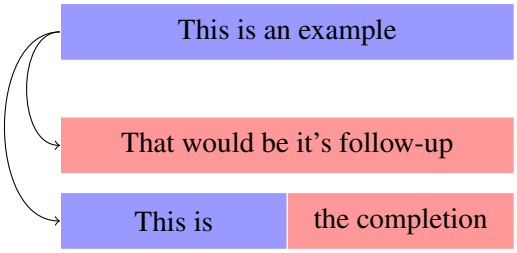

Figure 2: Generation methods examples. In blue is the initial example and in red is the generated text. The first generated example is from sentence follow-up, and the second is from sentence completion.

tri-trained models against the performances of a single model trained on the same amount of labeled natural data.

We do not reduce the size of our testing sets as we aim to compare our method to existing results. Our evaluation is comparative between our tri-training method and no augmentation method. We want to see whether there are increases in performance in a low-resource setting. Comparisons are made between a tagger trained on one subset against the ensemble of taggers obtained via our tri-training and generation method on the same subset. The sampling of subsets is seeded as explained in Section 4.2. We average results over those seeds to reduce the impact of selection biases.

4 Experiments

In this section, we describe the technical details of the experiments and explain the variants tested.

4.1 Datasets

The task we are working on is the Named Entity Recognition (NER) task. The goal of this task is to find mentions associated with certain concepts in sequences of text. In practice, this is done by assigning labels representing the concepts and the position within the mention to each of the tokens of the text. The corpora we are using are CoNLL 2003 English (Sang and De Meulder, 2003) and I2B2 (Uzuner et al., 2011). CoNLL is a corpus of Reuters news annotated with four different concepts: person, location, organization, and miscellaneous. The difficulties of this corpus reside in the various types of information portrayed within. From geopolitical news to tables of sports results, the input format varies greatly. I2B2 is a corpus of medical records annotated with three different concepts: problem, treatment, and test. These corpora are classic corpora for the NER task and cover

	I2B2	CoNLL
Train line count	11482	14986
Test line count	27625	3683
BERT base	84.0	90.0
BERT large	85.0	**92.0**
BioBERT	**86.6**	

Table 1: Reference models used as topline for our work and viability check against current state of the art. F_1 of BERT + classifier models on I2B2 and CoNLL using different pre-trained models. Metrics computed by seqeval (Ramshaw and Marcus, 1995) (Nakayama, 2018). Best model based on development set F1, trained on 50 epochs, with batch size of 32.

diverse specialty domains. These complete datasets contain enough data to be considered an ideal case for their respective tasks. We have tested our tagger architecture (see 4.3) on the full-sized data in order to verify its quality and select the best pre-trained BERT model available. This topline can be seen in Table 1. Our experiment focuses on low resources; the maximum size of the training data is less than 10% of the full set. We do not expect to reach topline results with our method at this quantity of data. However, we have to look at how much of the gap between topline and baseline is bridged by our method.

4.2 Low resource setting

The purpose of our method is to be used in a low-resource setting. We simulate such a setting by sampling a small number of labeled examples from the training set to create a new training set. We also consider that the quantity of data is small enough that all of the data is labeled. For our experiment, we reduce the training set to a subset S_{1000} of size 1000 by sampling without replacement using ten different seeds. This is where the sampling bias is induced. S_{1000} contains less than 10% of each of our sets. The seeding is done to reduce the variability of results due to sampling biases. Most of the results will be averaged over the ten seeds. We cut each subset S_{1000} in a series of subsets: $S_{50} \subset S_{100} \subset S_{250} \subset S_{500} \subset S_{1000}$. This is useful to evaluate the impact of the addition of new examples. For each seed, we obtain five subsets of labeled data.

4.3 Tagger

This section presents the architecture shared by all the taggers we train. It is a simple BERT +

		S_{50}	S_{100}	S_{250}	S_{500}	S_{1000}
I2B2	baseline	36.23±5.80	49.22±3.23	64.34±1.43	71.39±0.75	77.38±0.64
	Δunique	+3.93±1.89	+2.56±2.37	+1.89±1.25	+1.93±0.70	+1.28±0.84
	Δensemble	+4.32±1.82	+3.08±2.38	+2.45±1.23	+2.49±0.73	+1.80±0.84
CoNLL	baseline	59.87±3.32	69.20±3.92	80.65±1.99	84.74±0.89	87.70±0.38
	Δunique	+2.33±2.01	+0.08±3.64	+1.06±1.11	+0.54±0.83	+0.27±0.37
	Δensemble	+2.98±1.98	+0.84±3.68	+1.77±1.17	+1.14±0.71	+0.71±0.40

Table 2: F_1 score on baseline averaged across seeds. Average of the deltas between the performances of each individual tri-trained tagger and their respective baselines at Δunique lines. Average of the deltas between the performances of tri-trained ensembles and their respective baselines at Δensemble lines. Corpora used are I2B2 and CoNLL.

classifier architecture. The classifier is a two-layer feed-forward network with a hidden size of 768 and ReLU (rectified linear unit) activation. Dropout with $p = 0.1$ is applied between BERT and the classifier during training. The model is trained with the Adam optimizer with an initial learning rate of 10^{-5}. We train all taggers for tri-training and baseline for 1000 epochs with early stopping when the development set F_1 score stops increasing for 20 epochs (40 epochs for a subset of size 50). The sentence batch size is 16.

While we refer to our tagger architecture as BERT + classifier, we have tried different pre-trained BERT models[2][3][4] as shown in Table 1 and have settled on two different models. For CoNLL, the best results were obtained with BERT large cased (Devlin et al., 2018), and for I2B2, with BioBERT base cased (Lee et al., 2020).

4.4 Generation

We generate the unlabeled set U with GPT-2 (Radford et al., 2019). We use HuggingFace's implementation[5]. The text from the labeled train set is used as the context to generate entailed examples. With each labeled example, we generate five follow-up sentences. We also use the language model for sentence completion. In this case, we cut the original text and complete it using the model. Each labeled example is cut to 75%, 50%, and 25% of its length. In each of these cases, we generate five completed sentences. This amounts to a total of 20 synthetic examples per natural example. It is, in practice, slightly less than that because we filter out sequences made exclusively of different types of whitespace, newlines, and other such noise. Generated examples can be seen in Figure 3

4.5 Tri-training

The main focus of this article is the use of tri-training without natural unlabeled data. We use the unlabeled data generated, as explained previously, as the unlabeled data of tri-training. Tri-training requires one development set and one validation set: the first for the training of each model m_i, the second to validate the stagnation of the models across episodes. We chose to split the corpora's initial development set in half to fulfill each of those purposes. As this is a first experiment, we exclude sentences without tags from the pseudo-labeled set. This is done to avoid a possible problem at very low resources where the pre-trained models are not trained enough and produce sentences with empty tag sequences where they should not. However, our results show that these precautions might not be necessary. The result of the tri-training procedure is an ensemble of three models. Inference using this ensemble is done with a simple voting system. Voting is done by summing the scores output of each tag across all models and picking the highest.

4.6 Results

In this section, we present the results obtained across the different subsets.

Baseline. Baseline are the results of models trained in a supervised manner only on the natural training data. For each subset S_n, it is an average of 10 scores. The results in Table 2 show consistent performance increases between each subset sizes. Seqeval (Ramshaw and Marcus, 1995) (Nakayama, 2018) is used to compute the results. I2B2 $F1$ range from 36.2 (size 50) to 77.4 (size 1000), and

[2] https://huggingface.co/bert-base-uncased

[3] https://huggingface.co/bert-large-cased

[4] https://huggingface.co/dmis-lab/biobert-base-cased-v1.1

[5] https://huggingface.co/gpt2

ORG
MDS was founded in 1978.

PER
And it was then that Jussi Graf's

MISC
3_x86_64.tar.gz"") ; // We'll add this [...]

problem
FOLLOW US ON TWITTER!

test treatment
Disease tolerance test for benz

treatment
-12 10:27:28] RavenQueen > she's been so [...]

Figure 3: Examples of generation. The three first examples are from CoNLL and the three last from I2B2. Each series is formed of an example of completion and two examples of sentence follow-up. The examples were cherry-picked to show both positive and negative aspects of generation, be of short length, and be labeled by the models. On CoNLL's completion example, only a full stop was added. On I2B2's completion example, the context was "FOLLOW" and was too short and generic to bring the sentence to the medical domain. The second examples for both corpora are okay. The third examples for both corpora happen when short formulaic sentences are used as context. For CoNLL, it is the common -DOCSTART- and for I2B2, it was a date.

CoNLL $F1$ range from 59.9 (size 50) to 87.7 (size 1000). As discussed in Section 3.3, smaller sizes show a higher standard deviation with 5.8 for I2B2 and 3.3 for CoNLL at size 50.

Δunique. Tri-training produces three trained models supposed to be used as an ensemble of models. With constraints such as memory consumption or inference time, one might want to use a single model for inference. For such cases, we have reported the results of single models. The Δunique results show the deltas between each of the three individual models m_i and the baseline. For each subset S_n, it is an average of 30 deltas.

Δensemble. The purpose of tri-training is to obtain an ensemble of three models. We report the results of the ensembles by computing the deltas between the performances of the ensembles and their respective baselines. These results can be found within Table 2 at the Δensemble line and in Figure 4.

Our method obtains higher results on average on all subsets and on both corpora. Generally, on

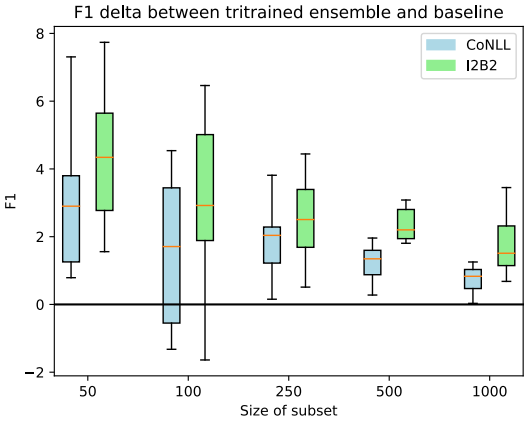

Figure 4: Boxplot of CoNLL and I2B2 deltas between tri-trained ensemble and baseline (Δensemble). For each subset size, the left boxplot is CoNLL, the right boxplot is I2B2.

I2B2, tri-training allows for a Δensemble to range from +4.32 (S_{50}) to +1.80 (S_{1000}). On CoNLL, it otherwise ranges from +2.98 (S_{50}) to +0.71 (S_{1000}). The Δunique shows, as expected, lower gains than Δensemble, ranging from +3.93 (S_{50}) to +1.28 (S_{1000}) for I2B2 and +2.33 (S_{50}) to +0.27 (S_{1000}) for CoNLL.

Out of the 50 individual runs for each corpus, one is negative for I2B2, and five are negative for CoNLL. Impacts of the negative results are seen on the average results of CoNLL at subset size 100. Three seeds yield negative gains at this size, with one having extreme (-8.6 points) negative gains. Removing this extreme result in the average calculation brings the Δensemble score closer to expected values (+1.89). Performances of individual models on CoNLL are within the standard deviation of negative results. This is not the case for I2B2. These results show that using the ensemble is a more stable solution. Overall, the method is most consistent with subsets of size 250 plus, as the average performance of tri-trained ensembles is above the standard deviation of the baseline.

5 Discussion

While our low-resource setting allows us to compare the impact of the training method in an otherwise similar context, it does not fully represent the nature of the problem. Building the development and test set is also a low resource problem. Reducing the test set to simulate low-resource will only make any comparison meaningless. Simulating the development set in the low resource context is an

improvement that could be made.

It is also to note that while the application domain is low resource, it is necessary to have a sizeable open-domain language model in the target language. Trying this method in languages other than English must be tested. Multilingual models might be the solution to the generalization of this method. As it stands, availability of large language model is the hardest limitation of this method.

6 Conclusion

Leveraging pre-trained models to improve performances on specific tasks is a common approach. With recent improvements to language modeling, recent models are often used directly to solve tasks. Direct usage is the method we use to build our taggers. However, we propose a new use for these sizeable models. They can serve as unlabeled data generators for semi-supervised learning. In particular, we have shown that we can use this method to gain significant improvements to the performances of taggers on NER and Clinical Concept Extraction in a low resource context. We gain between 3 and 4 points of F_1 score on subsets of data of size 50. Gains are overall positive on the sizes of the subsets we have tested. The higher the gains, the lower the data size is. We have shown that large language models are suitable tools to generate unlabeled examples for semi-supervised learning for NER.

Acknowledgements

This work was granted access to the HPC resources of IDRIS under the allocation 2021-AD011013018 made by GENCI. This work was granted access to the HPC resources of Saclay-IA through the Lab-IA machine. This work has been supported by the project PSPC AIDA: 2019-PSPC-09 funded by BPI-France.

References

Avrim Blum and Tom Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Y Chen, C Zhou, T Li, H Wu, X Zhao, K Ye, and J Liao. 2019. Named entity recognition from chinese adverse drug event reports with lexical feature based bilstm-crf and tri-training. *Journal of Biomedical Informatics*, 96:103252–103252.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Bosheng Ding, Linlin Liu, Lidong Bing, Canasai Kruengkrai, Thien Hai Nguyen, Shafiq Joty, Luo Si, and Chunyan Miao. 2020. Daga: Data augmentation with a generation approach for low-resource tagging tasks. *arXiv preprint arXiv:2011.01549*.

Thomas Dopierre, Christophe Gravier, and Wilfried Logerais. 2021. Protaugment: Intent detection meta-learning through unsupervised diverse paraphrasing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2454–2466.

Xuanli He, Islam Nassar, Jamie Kiros, Gholamreza Haffari, and Mohammad Norouzi. 2021. Generate, annotate, and learn: Generative models advance self-training and knowledge distillation. *arXiv preprint arXiv:2106.06168*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Citeseer.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

Antoine Neuraz, Leonardo Campillos Llanos, Anita Burgun, and Sophie Rosset. 2018. Natural language understanding for task oriented dialog in the biomedical domain in a low resources context. *arXiv preprint arXiv:1811.09417*.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

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*, 21:1–67.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora*.

Sebastian Ruder. 2019. *Neural Transfer Learning for Natural Language Processing*. Ph.D. thesis, National University of Ireland, Galway.

Sebastian Ruder and Barbara Plank. 2018. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1044–1054.

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2011. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association: JAMIA*, 18(5):552.

Jesper E Van Engelen and Holger H Hoos. 2020. A survey on semi-supervised learning. *Machine Learning*, 109(2):373–440.

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*, pages 5998–6008.

Wei Wang, Vincent W Zheng, Han Yu, and Chunyan Miao. 2019. A survey of zero-shot learning: Settings, methods, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–37.

Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *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 6382–6388, Hong Kong, China. Association for Computational Linguistics.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *33rd annual meeting of the association for computational linguistics*, pages 189–196.

Zhi-Hua Zhou and Ming Li. 2005. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering*, 17(11):1529–1541.