Label Semantic Aware Pre-training for Few-shot Text Classification

Veer Singh
University of Massachusetts
veersingh@umass.edu

Venkata Sai Phanindra
University of Massachusetts
vpamidimukkala@umass.edu

Lokesh Tangella
University of Massachusetts
ltangella@umass.edu

Sanjana Radhakrishna
University of Massachusetts
sanjanaradha@umass.edu

1 Introduction

The advancement of large pre-trained language models has greatly enhanced the performance of natural language processing (NLP) tasks, particularly in scenarios with limited training data. However, while these models excel at encoding input information, there has been limited investigation into providing informative representations of labels to the models. Most text classification methods commonly utilize label indices, but research suggests that employing sequence-to-sequence models to generate labels can improve performance. Nonetheless, these generative models only leverage label semantics during fine-tuning and prediction, rather than during pre-training.

To address this gap, a recent study introduced the Label Semantic Aware pre-training (LSAP) approach (A. Mueller and Roth, 2022), which integrates label semantics and input-label associations into the pre-training phase. This approach has shown the potential to achieve superior performance with a reduced number of fine-tuning examples across diverse domains. Our research aims to assess the effectiveness of LSAP by pre-training two generative models, whereas one incorporates label semantics and one does not. We will evaluate their few-shot performance on intent and topic classification datasets. Furthermore, time permitting, we plan to extend this approach to encoder models like BERT-base and evaluate their performance on sentiment analysis tasks.

2 Proposed & Accomplished Goals

Our principal aim is to implement the LSAP technique on the T5-small model and evaluate its performance across diverse few-shot settings, comparing it to a our baseline models. Furthermore, we have incorporated a stretch goal into our project, alongside our initial proposal. All the planned experiments have been successfully conducted, and the code can be found on GitHub[1]. Although we have made strides towards accomplishing the stretch goal, it remains incomplete as of now.

In the course of this investigation, we undertook a variety of tasks to examine the effectiveness of different pre-training approaches and fine-tuning strategies on downstream performance. The breakdown of our efforts is as follows:

- We gathered and preprocessed appropriate datasets for our study (refer to Section 4).
- We conducted a thorough exploration of three distinct pre-training formats, scrutinizing their performance under various shot settings, namely $1, 2, 4, 8, 16$-shot, and a full-resource scenario for downstream tasks.
- We implemented the Label Semantic Aware Pre-training (LSAP) technique and evaluated its efficiency through secondary pre-training applied on a quartet of T5-small models.
- Fine-tuning operations were undertaken with respect to hyperparameters, specifically for the downstream task of Intent Classification.
- We undertook the evaluation of the primary pre-trained T5-small model and the four secondarily pre-trained models under several few-shot scenarios, spanning from 1-shot to full-resource.
- Errors committed by the superior model in k-shot scenarios were scrutinized to gain a

[1] https://github.com/DigitalVeer/nlp-project

deeper understanding of the model's weaknesses.

- We made a comparative analysis of different metrics across all five models in the *k*-shot settings, aiming for a holistic understanding of their performance.

Initially, our study had a broader scope that included an ambitious stretch goal. However, we encountered limitations regarding time and computational resources that prevented us from achieving this goal. We were operating on Google Colaboratory, which offers limited GPU availability, thus constraining our computational capabilities. The unattained stretch goal involved an in-depth analysis of the effectiveness of the LSAP technique when applied to encoder-only models, like BERT, for a downstream task of sentiment analysis.

3 Related Work

Label semantics have been widely utilized to enhance performance and robustness in various settings and tasks, even prior to the prevalence of dense embedding representations. For instance, Chang et al. (Chang et al., 2008) achieved over 80% accuracy in binary text classification tasks without any labeled training examples by employing naive Bayes classifiers that utilized rich semantic representations of labels (Gabrilovich and Markovitch, 2007). Similarly, Song et al. (Song and Roth, 2014) adopted a similar strategy for hierarchical and multinomial classification tasks and found that dataless approaches could approach, and sometimes even outperform, supervised approaches in terms of performance.

More recently, label semantics based on dense embeddings have gained popularity, particularly with the advent of contextualized word embeddings (Peters et al., 2018). Label-wise attention networks (LWAN) have been proposed as one line of research for datasets with large structured label spaces, where multiple labels can apply to a long document. For example, Mullenbach et al. (Mullenbach et al., 2018) and Rios and Kavuluru (**?**) utilized LWANs based on convolutional neural networks and extended the attention mechanism for zero-shot settings. Chalkidis et al. (Chalkidis et al., 2020) employed BERT to embed the labels for an LWAN. In contrast, our setting involves a smaller label space and shorter input texts on average. Another label-semantic-aware approach, CLESS, performs contrastive pre-training on a question-answering dataset with a large and sparse label space for massively multi-label text classification. However, our setting exhibits a larger discrepancy between pre-training and fine-tuning, and we aim for more domain-generalization by leveraging token and span reconstruction objectives.

In the context of short-text intent and topic classification, other works have integrated label embeddings into their systems, similar to our task. For example, Gaonkar et al. (Gaonkar et al., 2020) and Ma et al. (Ma et al., 2022) used label embeddings from BERT and a label attention mechanism to improve emotion classification accuracy and few-shot named entity recognition, respectively. Generative approaches, such as those proposed by Rongali et al. (Rongali et al., 2020), Athiwaratkun et al. (Athiwaratkun et al., 2020a), and Paolini et al. (Paolini et al., 2021), implicitly leverage label semantics for text and token classification tasks by generating labels at prediction time.

Few-shot text classification is an important task in natural language processing that involves classifying text with only a few examples from the training set. Recent approaches include (Ro)BERT(a)-based methods (Chen et al., 2020) (Yang et al., 2019), prototypical networks, dynamic memory induction networks, and generative classification. In our work, we evaluate the LSAP approach on two specific subtasks of text classification: topic classification (TC) and intent classification (IC). In IC, the input is a conversational utterance, and the output is the label representing the user's intended action. Several recent few-shot IC approaches include dual encoders (Cer et al., 2018), Henderson et al. (Henderson et al., 2020), Casanueva et al. (Casanueva et al., 2020), Krone et al. (Krone et al., 2020) with prototypical networks and meta-learning, Zhang et al. (Zhang et al., 2020) with a nearest-neighbor discriminative method, and Yu et al. (Yu et al., 2021) with span-level contextualized embedding retrieval.

Generative text classification has become more feasible with recent advancements in large pre-trained language models, such as GPT (Radford et al., 2019) and GPT-3 (Brown et al., 2020). This approach involves training a model to generate natural language labels based on an input se-

quence, thereby reducing the train-test gap without requiring changes to the model's architecture. Pattern-exploiting training (PET) involves formatting training and testing examples as cloze-style prompts, where the label is typically a single word. PET enables a language model to see the inputs and embeddings of the output classes during tuning and prediction. Our approach differs in that we provide the model with concatenated utterances and labels, allowing it to transduce variable-length label sequences without reformatting the evaluation data. Seq2seq approaches using pointer/copy mechanisms (Rongali et al., 2020) or T5 (Raffel et al., 2020) have shown effectiveness and data efficiency in text and token classification tasks, such as slot labeling (Athiwaratkun et al., 2020b) and entity-relation extraction (Paolini et al., 2021). In our approach, we build upon T5-small but differ from prior work by performing a label-semantic-aware secondary pre-training step on a variety of datasets before fine-tuning.

4 Data

Our pre-training data consists of utterances with intent labels (Table 1), similar to the original LSAP paper. For pre-training, we utilize the same dataset that the authors do. In particular, our pre-training corpus is created by combining the PolyAI Banking dataset[2] and the WikiHow intents dataset[3]. The PolyAI Banking dataset contains over 10,000 training examples categorized into 77 unique intents, focusing on online banking queries.

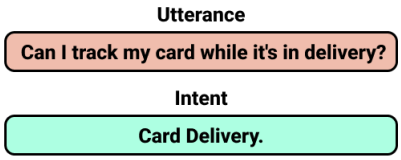

Figure 1: Example Utterance from Polybank

The WikiHow intents dataset consists of heuristically labeled examples derived from the WikiHow articles. In this dataset, each example consists of a single line taken from a WikiHow article, typically representing the longest step in the article. The intent label for each is derived from the article title by removing the phrase "How To".

[2]The PolyAI Banking dataset is available at https://polyai.com/datasets/banking.

[3]The WikiHow intents dataset is available at https://github.com/zharry29/wikihow-intent.

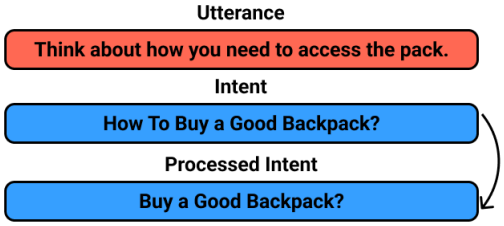

Figure 2: Processing Intents from WikiHow

The pre-training corpus is created by combining these two datasets, resulting in a corpus size of a little over 120,000 examples.

For evaluation of the models, we used samples from ATIS[4], Snips[5], TOPv2(remainder), and TOPv2(weather) datasets. The ATIS dataset is commonly used for benchmarking intent classification models and TOPv2 is common for task-oriented semantic parsing. These datasets served as benchmark datasets for our downstream task of intent classification. The Snips dataset and ATIS datasets are balanced, while TOPv2 dataset is unbalanced. We specifically chose these datasets as they are non-overlapping with our pre-training corpus. All the datasets used in this project are publicly available, and their sources are provided in the footnotes.

We categorize the quality of the PolyBank dataset as 'gold' and the WikiHow dataset as 'silver' based on the method used to label the intents. The PolyBank dataset is human-labeled, and similar to the authors, we infer its quality as gold. On the other hand, the WikiHow dataset is labeled heuristically, leading us to categorize its quality as silver.

4.1 Data preprocessing

In the LSAP approach, a secondary pre-training is performed using the data labeled with semantics. The following pre-training formats are considered/evaluated:

1. **Random Span Denoising:** One pre-training objective is to use random span denoising. In this approach, each intent is appended to the end of it's respective utterance. At this point,

[4]The ATIS dataset is available at https://github.com/howl-anderson/ATIS_dataset/.

[5]The Snips dataset is available at https://github.com/ZhenwenZhang/Slot_Filling.

Dataset	Quality	Description	Training Examples	Intents
PolyAI Banking	Gold	Online banking queries.	10,003	77
WikiHow Intents	Silver	WikiHow Article Titles with "How To" removed. Each utterance is the longest instruction in the article.	110,000	66,343

Table 1: Pre-training datasets. "Quality" refers to the source of the labels (human-labeled is gold, deterministically labelled is silver)

15% of the tokens in the utterance-intent input sequence are randomly masked, with the goal to reconstruct the noised spans in the output sequence. This is the same objective that T5 uses.

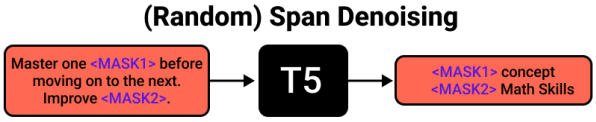

Figure 3: Random Span Denoising

2. **Intent Classification:** In intent classification (IC) pre-training, we prefix "intent classification: " to the start of every utterance, and keep the output sequence as the intent. The idea is to keep the data in the same format used for supervised fine-tuning.

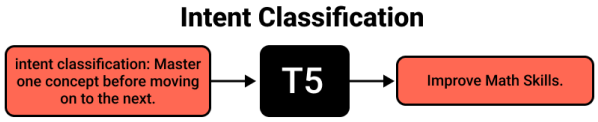

Figure 4: Intent Classification

3. **Label Denoising:** In Label Denoising, we always noise the entire intent label, and append a mask to the end of each utterance. The pre-training objective is to reconstruct the output sequence from the given mask. As the authors mention, this is a way of framing the intent classification task as an unsupervised denoising task.

As our results show, the choice of pre-training used has a huge impact on our project. We implement different n-shot models that are used on all three of our pre-training formats and concur with the authors that label denoising is the preferred pre-training format for our downstream task.

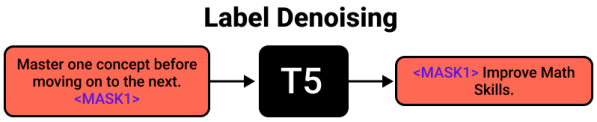

Figure 5: Label Denoising

5 Baselines

Our baselines consist of three models:

- **Model 1:** T5-Small with no secondary pre-training.
- **Model 2:** T5-Small with secondary pre-training on utterances.
- **Model 3:** T5-Small with secondary pre-training on utterances along with their intents.

We conduct evaluations of these three models across multiple few-shot learning settings, namely 1-shot, 2-shot, 4-shot, 16-shot, and full resource settings. In this study, a *k-shot* setting denotes the process of fine-tuning a pre-trained model using k samples per label, which includes model parameter updates. Conversely, the full resource setting entails fine-tuning on the entirety of the available labeled data for the downstream task.

Considering that Model 3 has undergone pre-training with intents, we anticipate its favorable adaptation to the downstream intent classification task even with limited labeled examples for each label. As a result, we expect Model 3 to outperform both T5-Small and Model 2, establishing it as our superior model. T5-Small, lacking secondary pre-training, typically necessitates a larger amount of data for effective fine-tuning in downstream tasks. Similarly, Model 2, pre-trained without intents, lags behind Model 3 in few-shot settings for the intent classification task. Hence, T5-Small and Model 2 are selected as our baseline models.

Moreover, the performance order of these models generally follows Model 1 $<$ Model 2 $<$ Model 3 in the k-shot setting. However, as the value of k increases, the performance of all models tends to converge, as they receive more data to adapt to the intent classification task during fine-tuning.

Additionally, we have conducted experiments with two additional T5-Small based models, referred to as Model 4 and Model 5. Both of these models are initialized with random weights, effectively erasing the domain-specific knowledge obtained from pre-training. Subsequently, Model 4 and Model 5 are pre-trained using our pre-training data (discussed in Section 4), following the same approach as Model 2 and Model 3, respectively. Model 5, pre-trained with intents, is expected to outperform Model 4, similar to the comparison between Model 2 and Model 3. Therefore, we consider Model 4 as the baseline for Model 5. It is worth noting that T5-Small is still expected to outperform Model 5 in this scenario due to its pre-training on a large corpus.

Our experiments involved various datasets (discussed in Section 4), which were utilized for secondary pre-training and fine-tuning in the k-shot learning setting. During the pre-training phase, the Poly AI Banking and WikiHow datasets were split into training and validation sets in a 96:4 ratio, albeit an 80:20 split being ideal. The choice of the 96:4 split was due to computational resource limitations. The validation set played a role in computing metrics such as perplexity after each epoch during the secondary pre-training phase. Conversely, the datasets used for fine-tuning typically followed a standard 90:5:5 split for train/test/validation sets. Stratification was applied to ensure equal proportions of labels in these sets.

5.1 Hyperparameter Tuning

While the pre-training and fine-tuning phases contribute to learning the parameters of the NLP model, the performance and learning rate are governed by hyperparameters. Tuning these hyperparameters is crucial to achieve the desired results in the models.

During the secondary pre-training stage, we carefully adjusted hyperparameters, including the learning rate, weight decay, number of training epochs, training batch size, and evaluation batch size, following the guidelines provided in the referenced paper. However, due to resource limitations, we encountered out-of-memory issues. To address this, we experimented with reducing the batch size from the initial 128 to various ranges, eventually reaching a batch size of 1. While this allowed the model to execute, it significantly increased the training time. Moreover, when we slightly increased the evaluation batch size, we faced memory issues once again. To overcome this, we employed gradient accumulation with a factor of 4, utilizing an evaluation batch size of 2 and a training batch size of 8. This approach enabled the model to complete the training epochs without memory constraints. Throughout the training process, we logged perplexity and loss values for both the training and evaluation datasets to gain insights into the model's progress.

Following further heuristic adjustments, we settled on the following hyperparameter configuration: a training batch size of 16, an evaluation batch size of 2, gradient accumulation steps of 4, a learning rate of 5×10^{-3}, and a total of 5 training epochs. With these hyperparameters, all models achieved train and validation perplexity values within the range of 1-2, except for Model 4 and Model 5. These two models exhibited perplexity values above 40 at the end of the fifth epoch. To improve their performance, we extended the training duration to 10 epochs, resulting in a reduction of perplexity to approximately 20. It is important to note that the higher perplexity of Model 4 and Model 5 can be attributed to their initialization with random weights, causing the loss of some information from the large pre-training corpus.

In the fine-tuning stage or few-shot learning stage, we have performed hyperparameter tuning, including learning rate, weight decay, number of training epochs, training batch size, and evaluation batch size. Due to limited computational resources and the time-consuming nature of each run, we manually adjusted these hyperparameters one at a time. During the adjustment process, the model was trained and its performance on the validation set was evaluated. The combination of hyperparameters leading to the best metric (accuracy in our case) was ultimately chosen for evaluating the model on the test set. Hyperparameter tuning is a tedious task, especially considering the large number of possible combinations. To streamline the process, we kept all hyperparameters fixed except for one, observed the direction of change in

the accuracy of the validation set, and adjusted the corresponding hyperparameter accordingly. This iterative process was repeated until satisfactory results were achieved. Hyperparameter tuning was performed for all our models, and the test set was kept hidden from the model until all hyperparameters were tuned and the final model was ready after pre-training and few-shot learning.

The hyperparameters used varied across different models and datasets. The typical ranges we considered were as follows:

1. Learning Rate: Typically chosen from the range $[4 \times 10^{-3}, 2 \times 10^{-2}]$
2. Batch Size: Chosen from $\{16, 32\}$
3. Weight Decay: Set to 0
4. Number of epochs: Either 3 or 10 (varied for different datasets)

6 Approach

Previous research in related works has explored the incorporation of supplementary details, such as the intent of input sequences, into models. However, these approaches typically introduce this additional information during the fine-tuning phase, where domain-specific data is provided along with the corresponding intents. Little to no work has been done on utilizing label semantics during the pre-training phase. In this paper, we propose a novel approach called Label Semantic Aware Pre-training (LSAP), which leverages the valuable label semantics during the pre-training phase to achieve improved performance in few-shot settings.

The goal of our project is to analyze whether the few-shot performance of a language model improves when intents (in the case of intent classification as the downstream task) are introduced during a secondary pre-training stage. To evaluate this hypothesis, we have considered five main models, including the baseline models, all based on the T5-Small model with 60 million parameters.

The five models considered in our study are as follows:

1. **Model 1:** *T5-Small*: A pre-trained T5-Small model.
2. **Model 2:** *T5-Small-Adapted*: A pre-trained T5-Small model with secondary pre-training on our pre-training corpus (discussed in Section 4) without intent.
3. **Model 3:** *T5-Small-LSAP*: A pre-trained T5-Small model with secondary pre-training on our pre-training corpus along with intent labels.
4. **Model 4:** *T5-Small-Nomad-Adapted*: T5-Small model with randomly initialized weights and then pre-training using our pre-training corpus without intent labels.
5. **Model 5:** *T5-Small-Nomad-LSAP*: T5-Small model with randomly initialized weights and then pre-training using our pre-training corpus along with intent labels.

There are various ways to perform secondary pre-training for the aforementioned models (except for Model 1). Among these options, we have evaluated the following pre-training formats:

1. Label Denoising Format: This format involves denoising the labels during the secondary pre-training process.
2. Random Span Denoising Format: In this format, random spans are denoised during the secondary pre-training phase.
3. Intent Classification (IC) pre-training Format: This format includes pre-training the models using intent classification as an auxiliary task.

To determine the best pre-training format among these three options, we conducted secondary pre-training on three T5-Small-based models. It is worth noting that the label denoising format is the same as the other models:

1. **Model 6:** *T5-Small-Random-Span*: A pre-trained T5-Small model with secondary pre-training performed using the Random span denoising format.
2. **Model 7:** *T5-Small-IC-pre-train*: A pre-trained T5-Small model with secondary pre-training using the Intent Classification (IC) pre-training format.

Table 2: ATIS Fine-tuning Accuracies

Formats	one_shot	four_shot	full_resource
Model 3- label denoising	67.3	77.48	96.86
Model 6 - random span denoising	25.28	42.73	97.09
Model 7 - IC Pre-Train	3.58	46.76	97.54

Table 3: SNIPS Fine-Tuning Accuracies

Formats	one_shot	four_shot	full_resource
Model 3- label denoising	49.71	79	96.71
Model 6 - random span denoising	67.43	84.57	0.0
Model 7 - IC Pre-Train	42.0	80.14	0.0

Table 4: TOPS Weather Fine-Tuning Results

Formats	one_shot	four_shot	full_resource
Model 3 - label denoising	95.54	51.71	0.0
Model 6 - random span denoising	72.4	79.9	0.0
Model 7 - IC Pre-Train	78.48	48.98	0.0

We conducted secondary pre-training on Model 3, Model 6, and Model 7 using our pre-training corpora (discussed in Section 4). Subsequently, we evaluated the performance of these models in 1-shot, 4-shot, and full-resource learning settings. The experiments were conducted on three different datasets: SNIPS, ATIS, and TOPS Weather datasets. The accuracy results on the respective test sets are presented in Tables 2 to 4.

From the results, we observe that Model 3, which underwent Label Denoising pre-training, consistently outperforms the other models in all three k-shot splits. However, it is worth mentioning that the full-resource accuracy for the Weather dataset is reported as zero due to a job cancellation during runtime, likely due to the large amount of data.

Intuitively, the superior performance of the Label Denoising pre-training format can be attributed to the fact that the model is trained to predict the masked label (intent) during the pre-training process itself. This serves as a form of supervision for intent classification, enhancing the model's ability to accurately classify intents.

Thus, we utilize the Label Denoising pre-training (secondary) format to pre-train Models 2, 3, 4, and 5, allowing us to evaluate the LSAP performance in various few-shot settings. For fine-tuning, we pre-train Models 2, 3, 4, and 5 on our pre-training corpora and evaluate their performance in $\{1, 2, 4, 8, 16\}$-*shot* settings, as well as finally in a full-resource environment on two datasets: SNIPS and ATIS. Additionally, we create data for 1-shot, 2-shot, ... 16-shot settings, where the data for P-shot is a subset of the data for N-shot, with $P < N$. The corresponding results of the various few-shot splits of ATIS and SNIPS for Model 1, Model 2, and Model 3 are shown in Table 5 and Table 6, respectively.

As discussed previously, Model 3 is our best model, and the results from Tables 2 and 3 are in line with our expectations. Although we couldn't achieve the same accuracy across our models as reported by the original authors, we attribute this to the choice of model. The authors of the paper used the T5 model with 3 billion parameters, whereas we used T5-small with only 60 million parameters due to limitations in computational power. Nonetheless, we can still observe the same trend of Model 3 outperforming our baselines (T5-small and Model 2). Furthermore, we notice that the performance difference between these models decreases as k increases in the k-shot setting, and they perform almost equally in the full-resource setting.

Tables 2 and 3 exclusively present the results

Table 5: ATIS Evaluation Accuracies

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	51.03	37.21	53.36	72.87	80.21	97.30
Model 2	50.6	66.0	47	38.11	82.28	96.18
Model 3	67.3	69.5	77.48	79.4	84.75	96.86

Table 6: SNIPS Evaluation Accuracies

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	51.85	59.42	75.42	86.28	91.71	97.28
Model 2	26.71	48	81.71	91.85	93.85	96.71
Model 3	49.71	77.71	79	93.85	94.85	96.71

for Models 1, 2, and 3, while the outcomes for Model 4 and Model 5 are not reported. These two models exhibited a 0 accuracy across all settings for intent classification. Although this outcome is unexpected, we attribute it to the limited size of our pre-training corpora. It should be noted that in Model 4 and Model 5, we initialize the weights randomly before conducting pre-training on our corpora. However, as our pre-training data consists of only 120K utterances + intents, it is inadequate to effectively pre-train a large-scale model like T5-Small with millions of parameters.

During the pre-training stage, both Model 4 and Model 5 exhibited a minimum perplexity of 10, in contrast to the perplexity of 1.5 achieved by Model 2 and Model 3. To improve their performance on the downstream task, we attempted to pre-train these models on the same dataset for a longer duration of 20 epochs. However, no notable enhancement was observed. Thus, we posit that due to the scarcity of pre-training data, these two models encountered challenges in generalization, resulting in poor performance on the downstream task. We further scrutinized the outputs of these models after fine-tuning on the full resource and discovered that they often classified nearly half of the input utterance as the intent, consequently yielding a zero accuracy.

6.1 Implementation Details

- We have imported T5-small model from huggingface library and have used various other classes and implementations from Transformers library in huggingface to pre-train, fine-tune, evaluate and test the model.
- We have used Datasets and Pandas library to work with the data in-memory.
- We have used Pytorch Library through out our code base.
- We have used numpy library for efficient matrix operation wherever necessary.

A major portion of our experiments was conducted on Google Colab. As is known, Google Colab has a hardware limitation of 4 hours per user per day. This resulted in various issues, especially during model pre-training. The pre-training process would abort once the allocated limit was reached. Furthermore, the issue exacerbated during hyperparameter tuning, where we needed to try various combinations of hyperparameters to find the best settings. The same issue was observed during fine-tuning on the full resource. We had to save checkpoints regularly and then continue from a different account using these checkpoints to complete the task.

Towards the end, we ran some of our experiments on the Unity Server provided by UMass Amherst. Although the cluster does not impose usage time limits, the allocated memory (RAM) per individual is limited. During the pre-training stage, we frequently encountered memory out-of-bounds issues. That is why we pre-trained the model on a smaller dataset. This is the reason why we failed to achieve the expected results with Model 4 and Model 5.

6.2 Result Analysis

During the evaluation stage of our pipeline, we went beyond what the original authors devised for measuring the performance of the models. While

Table 7: ATIS Bleu Score Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	12.54	12.12	13.15	15.77	15.64	17.98
Model 2	2.94	11.87	4.50	13.28	16.12	18.24
Model 3	0.06	8.71	3.61	8.47	14.75	18.21
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 8: ATIS Cosine Similarity Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	0.83	0.86	0.89	0.95	0.91	0.98
Model 2	0.26	0.83	0.77	0.89	0.94	0.99
Model 3	0.18	0.69	0.54	0.87	0.94	0.99
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 9: First Word Accuracy Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	68.9	64.65	71.59	85.23	83.67	96.42
Model 2	1.34	66.89	24.38	71.14	86.58	97.76
Model 3	0.0	49.22	16.55	43.62	78.75	97.76
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 10: ATIS Jaccard Score Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	67.19	64.28	70.32	84.60	83.37	96.27
Model 2	9.51	66.29	22.31	70.30	85.90	97.61
Model 3	0.22	48.32	16.82	43.43	78.30	97.61
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 11: SNIPS Bleu Score Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	23.61	31.52	36.03	36.61	39.02	39.80
Model 2	18.74	29.16	30.81	37.75	39.07	0.0
Model 3	13.29	31.03	33.33	0.0	39.44	0.0
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

the authors used accuracy as their primary metric, we decided it would be worth inferring other metrics about the quality of our models as well. In addition to accuracy, we consider the following metrics:

1. **Cosine Similarity**: We examine the similarity between two intents by calculating the co-

Table 12: SNIPS Cosine Similarity Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	0.57	0.74	0.87	0.86	0.92	0.95
Model 2	0.44	0.71	0.73	0.89	0.93	0.0
Model 3	0.27	0.73	0.78	0.0	0.93	0.0
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 13: SNIPS First Word Accuracy

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	61.86	83.86	92.86	92.0	95.14	96.43
Model 2	55.57	74.29	82.43	92.0	94.43	0.0
Model 3	35.29	81.57	82.71	0.0	94.57	0.0
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

Table 14: SNIPS Jaccard Score Results

	one_shot	two_shot	four_shot	eight_shot	sixteen_shot	full_resource
Model 1	58.99	76.27	87.82	87.54	92.97	95.29
Model 2	47.92	72.51	75.06	89.35	93.17	0.0
Model 3	28.10	75.23	79.42	0.0	93.66	0.0
Model 4	0.0	0.0	0.0	0.0	0.0	0.0
Model 5	0.0	0.0	0.0	0.0	0.0	0.0

sine of the angle between them in embedding space. This holds an advantage over simply using accuracy because it allows us to measure if the model is predicting semantically similar intents to our current intent, along with a measurement of the approximate closeness. (Figure 6)

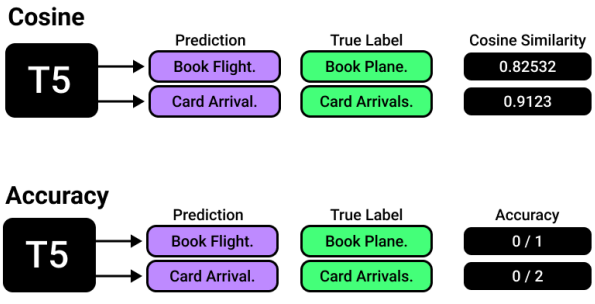

Figure 6: Cosine Similarity allows us to examine predictions in a way that accuracy cannot.

2. **BLEU Score**: This metric evaluates the quality of generated text by comparing it against one or more reference texts. It measures the precision of n-grams in the generated text. Here, we use it to examine n-gram ($n = 2$) comparisons between the truth and prediction labels. In the case where our generation may have an extra word or two, we can catch this (Figure 8).

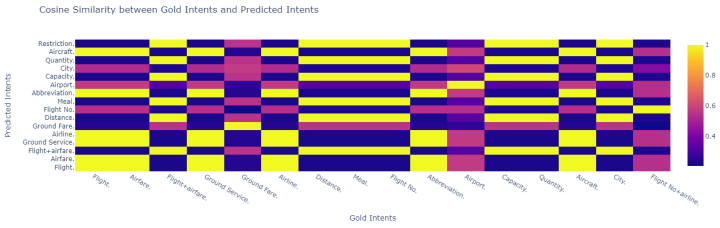

Figure 7: Using cosine similarity, we can also see which predictions are closest to which ground truths with a heatmap.

3. **Jaccard Similarity**: This metric quantifies the similarity between two sets by measuring the intersection over the union of the sets. It provides insight into the overlap and common

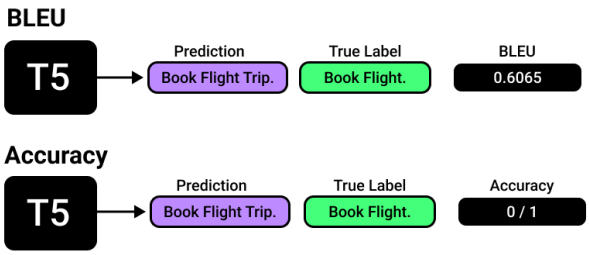

Figure 8: BLEU is beneficial over accuracy for evaluating generated text because it considers the precision of n-gram sequences, allowing for slight variations and capturing semantic similarity between the generated and reference texts.

elements between the predicted and ground truth sets. We expect Jaccard and BLEU to be similar in most cases when using $n = 2$ for BLEU; however, Jaccard Similarity let's us evaluate how many words in the true intent the model was about to generate. This again let's us catch when the model is very close, but not guessing the exact truth label.

These metrics helped us look at our models with a bit more depth than the original authors were able to. We are able to see that Model 1 consistently demonstrates superior performance across all evaluation measures for the ATIS dataset. It exhibits high accuracy, BLEU scores, cosine similarity, and Jaccard similarity, indicating accurate intent prediction and semantically similar text generation compared to the ground truth labels. However, Model 2 and Model 3 exhibit relatively poor performance with lower accuracy, BLEU scores, and Jaccard similarity, suggesting challenges in predicting correct intents and generating text that aligns well with the ground truth labels. Lastly, model 4 and model 5 both perform poorly with a 0 across most metrics.

While Model 2 and Model 3 demonstrate reasonably good performance in terms of accuracy, BLEU scores, and Jaccard similarity, their cosine similarity is lower compared to Model 1, indicating that their generated intents and text may lack semantic similarity with the ground truth labels. Furthermore, the addition of other metrics allow us to examine that Model 4 and Model 5 both in more detail. For example, they have 0 accuracy across all n-shot settings; however, are they close? We can see from our other metrics that these two models are not close in any aspect; neither semantically or lexicographically.

7 Error Analysis

Our focus was on few-shot learning, and particularly, we sought to investigate how our model grasped and processed sentences when trained on a 4-shot SNIPS dataset. Out of approximately 120 examples, we found that 80% of the sentences were incorrectly predicted as either "SearchScreeningEvent" or "SearchScreeningEvent". This could be attributed to the biasing of the first word in the label as the model's (Model 3) initial prediction, since we employed white space for word splitting. With a scant four examples for training, the model seemed unable to glean intricate details of the intent.

The following list outlines the various intents contained within the SNIPS dataset:

- "SearchCreativeWork" (e.g. Find me the I, Robot television show)
- "GetWeather" (e.g. Is it windy in Boston, MA right now?)
- "BookRestaurant" (e.g. I want to book a highly rated restaurant in Paris tomorrow night)
- "PlayMusic" (e.g. Play the last track from Beyoncé off Spotify)
- "AddToPlaylist" (e.g. Add Diamonds to my roadtrip playlist)
- "RateBook" (e.g. Give 6 stars to Of Mice and Men)
- "SearchScreeningEvent"

Upon examination of the examples, it became apparent that the model was prone to confusion when a location was specified in the text. An illustrative example is, "what is the forecast for Ōtone Prefectural Natural Park in 1 hour and within the same area". As the model predicted "RateBook" by overlooking the word 'forecast', it suggested that it was misconstruing the intent as booking-related due to the presence of a location.

This tendency was also observed in examples like, "what is the forecast for Costa Rica", "Tell me the forecast for 6 am in Tatra-Nationalpark". It was thus evident that the model, trained with only 4-shot examples, struggled to comprehend the "GetWeather" and "PlayMusic" labels, and was

more biased towards "SearchScreeningEvent", "RateBook" and "SearchCreativeWork".

Further confusions were noted with search-related intents such as, "Find the album Orphan Girl at the Cemetery", where the model was expected to predict "SearchCreativeWork" but instead produced "SearchScreeningEvent".

In addition, the model grappled with sentences containing the word 'music', even though "PlayMusic" and "SearchCreativeWork" are intrinsically related. An example includes, "Let's hear good Mohammad Mamle on Vimeo".

To gain a more detailed understanding of the model's performance, we escalated to 8-shot training, which significantly improved the model's predictions, reducing errors to 49. Consequently, the model shed its bias towards "RateBook" and was able to distinguish between "SearchCreativeWork" and "SearchScreeningEvent", thus trimming the number of erroneous predictions by 12 in subsequent evaluations.

8 Conclusion

Our exploration of the Label Semantic Awware Pretraining (LSAP) technique has brought forth intriguing insights that underline its generic applicability and efficacy in enhancing performance within few-shot settings. A key observation from our study is the demonstrable capability of large language models to perform exceptionally well with the minimal guidance of a single downstream task example, as long as they are equipped with domain general knowledge. This revelation paves the way for contemplating the far-reaching potential of large language models in the foreseeable future. We noted that the pretraining process was more challenging than anticipated. Specifically, the requirement of large data volumes and substantial computational power to effectuate the randomization of existing weights and to kick-start pretraining from scratch posed significant hurdles.

Looking ahead, given sufficient computational resources, our future work would focus on conducting a comparative study of our Models 4 and 5. Such a comparison would definitively illustrate the robustness of the LSAP technique, considering that the primary variance between these two models is the incorporation of intents during pretraining. Moreover, we plan to assess the versatility of the LSAP technique by applying it across diverse downstream tasks and a spectrum of language model architectures. This would allow us to further ascertain the broad-spectrum applicability of this innovative approach.

References

A. Mueller, J. Krone, S. R. S. M. E. M. Y. Z. and Roth, D. (2022). Label semantic aware pre-training for few-shot text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Athiwaratkun, B., dos Santos, C. N., Krone, J., and Xiang, B. (2020a). Augmented natural language for generative sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 375–385, Online. Association for Computational Linguistics.

Athiwaratkun, B., dos Santos, C. N., Krone, J., and Xiang, B. (2020b). Augmented natural language for generative sequence labeling. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 375–385, Online. Association for Computational Linguistics.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Casanueva, I., Temčinas, T., Gerz, D., Henderson, M., and Vulic, I. (2020). Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 38–45, Online. Association for Computational Linguistics.

Cer, D., Yang, Y., Kong, S.-y., Hua, N., Limtiaco, N., St. John, R., Constant, N., Guajardo-Cespedes, M., Yuan, S., Tar, C., Sung, Y.-H., Strope, B., and Kurzweil, R. (2018). Universal sentence encoder. *Computing Research Repository*, arXiv:1803.11175.

Chalkidis, I., Fergadiotis, M., Kotitsas, S., Malakasiotis, P., Aletras, N., and Androutsopoulos, I. (2020). An empirical study on large-scale multi-label text classification including few and zero-shot labels. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7503–7515, Online. Association for Computational Linguistics (ACL).

Chang, M.-W., Ratinov, L.-A., Roth, D., and Srikumar, V. (2008). Importance of semantic representation: Dataless classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 2, pages 830–835, Chicago, Illinois. Association for the Advancement of Artificial Intelligence (AAAI).

Chen, X., Ghoshal, A., Mehdad, Y., Zettlemoyer, L., and Gupta, S. (2020). Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Gabrilovich, E. and Markovitch, S. (2007). Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1606–1611.

Gaonkar, R., Kwon, H., Bastan, M., Balasubramanian, N., and Chambers, N. (2020). Modeling label semantics for predicting emotional reactions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4687–4692, Online. Association for Computational Linguistics (ACL).

Henderson, M., Casanueva, I., Mrkšić, N., Su, P.-H., Wen, T.-H., and Vulic, I. (2020). ConveRT: Efficient and accurate conversational representations from transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2161–2174, Online. Association for Computational Linguistics.

Krone, J., Zhang, Y., and Diab, M. (2020). Learning to classify intents and slot labels given a handful of examples. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 96–108, Online. Association for Computational Linguistics.

Ma, J., Ballesteros, M., Doss, S., Anubhai, R., Mallya, S., Al-Onaizan, Y., and Roth, D. (2022). Label semantics for few shot named entity recognition. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1065–1073, Dublin, Ireland. Association for Computational Linguistics.

Mullenbach, J., Wiegreffe, S., Duke, J., Sun, J., and Eisenstein, J. (2018). Explainable prediction of medical codes from clinical text. 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 1101–1111, New Orleans, Louisiana. Association for Computational Linguistics (ACL).

Paolini, G., Athiwaratkun, B., Krone, J., Ma, J., Achille, A., Anubhai, R., Santos, C. N. d., Xiang, B., and Soatto, S. (2021). Structured prediction as translation between augmented natural languages. In *International Conference on Learning Representations*.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics (ACL).

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Rongali, S., Soldaini, L., Monti, E., and Hamza, W. (2020). Don't parse, generate! a sequence to sequence architecture for task-oriented semantic parsing. In *Proceedings of The Web Conference 2020*, pages 2962–2968, New York, New York. Association for Computing Machinery.

Song, Y. and Roth, D. (2014). On dataless hierarchical text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, pages 1864–1870, Quebec City, Canada. Association for the Advancement of Artificial Intelligence (AAAI).

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32.

Yu, D., He, L., Zhang, Y., Du, X., Pasupat, P., and Li, Q. (2021). Few-shot intent classification and slot filling with retrieved examples. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 734–749, Online. Association for Computational Linguistics.

Zhang, J., Hashimoto, K., Liu, W., Wu, C.-S., Wan, Y., Yu, P., Socher, R., and Xiong, C. (2020). Discriminative nearest neighbor few-shot intent detection by transferring natural language inference. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5064–5082, Online. Association for Computational Linguistics.