TK-KNN: A Balanced Distance-Based Pseudo Labeling Approach for Semi-Supervised Intent Classification

Venkata Sai Ayyappa Hemanth Duddu
George Mason University
Computer Science
G01413649
vduddu@gmu.edu

Pranitha Kakumanu
George Mason University
Computer Science
G01379534
pkakuman@gmu.edu

Aadithya kiran Tulabandu
George Mason University
Computer Science
G01417888
atulaban@gmu.edu

1 Introduction

1.1 Task / Research Question Description

Considering the high use of virtual assistant systems for many businesses, these systems need to detect the correct intent in the dialogues or conversations. To attain the ability, there are many challenges encountered by these systems to understand the intention of the user. It requires a large data to learn and label the data, which is very expensive and time-consuming. Thus, the researchers use Semi-Supervised learning involving small labeled data to train the systems and predict the intent label for unlabelled data. But this approach comes with the risk, there is a set confidence threshold for the model, and a pseudo-label to the unlabeled data if their confidence is above this set threshold. Although this can improve the performance, it can also make the model overconfident for a few intents.

In this paper, the authors have introduced a new concept or method called Top-K K-Nearest Neighbor (TK-KNN). This method addresses the overconfidence risk by using a top-k sampling strategy ensuring a balanced set of classes and leveraging or influencing the embedding space of labeled and unlabelled data making more predictions accurately. In the paper, the authors have tested and experimented with different datasets, like CLINC150, BANKING77, and HWU64.

1.2 Motivation and Limitations of existing work

Motivation: In today's world of automated virtual customer service or virtual computer agents, a huge amount of unlabeled data is created which makes it very hard for humans to label the data manually. This data can be used to train a model to find the right intent of the customer to provide the best services. So, the motivation is to create a model that makes use of the context of the text and labels the data.
Previously, many attempts were made using pseudo labeling and distance-based pseudo labeling, but the main issue with these methods is the mislabeling of data in early iterations, which leads to a degradation in the overall performance of the system.

Limitations: (1) This paper doesn't require searching for a good confidence threshold, but this paper introduces two new hyperparameters k and β that need fine-tuning.
(2) The proposed self-training method requires more training cycles for effective labels, which requires a very high amount of GPU.
(3) This paper didn't explore heavily imbalanced datasets, so the performance of TK-KNN to those datasets is still a question mark.

1.3 Proposed Approach

Traditionally, the semi-supervised learning methods rely on pseudo labeling, where iterative unlabelled data is labeled by the model based on the prediction. However, these methods often struggle with biased labeling.

This paper's core contribution is introducing TK-KNN, which handles these challenges by introducing a more balanced and nuanced approach. TK-KNN combines the model's confidence scores with the cosine similarity based on tokens in embedding space between labeled and unlabelled data then we only take top k elements with the highest similarity score. Then we do this same procedure iteratively until all the data points are labeled. TK-KNN uses top-k sampling to select the most confident predictions from the model for each class and this helps in selecting the most reliable predictions and avoiding the potential noisy prediction. Using the KNN Search for TK-KNN incorporates the similarities between the unlabelled data and its nearest neighbors from the labeled data in embedding space. By considering the model's prediction and similarity, the TK-KNN model creates balanced decision boundaries for different intent classes, ensuring that all classes are spread evenly and represented effectively by reducing the risk of mislabeling and improving classification accuracy. Both, top-K sampling and KNN search allow TK-KNN to understand more discriminative features. This improves the performance among various intent classification tasks.

1.4 Checkpoint 2 Approach

1.4.1 Training a Multilingual Model

For checkpoint 2, we took the Multilinguality task. Initially, we have the three datasets, CLINC150, banking77, and HWU64 in English. For the classification task, the Author has used **BERT**, so we decided to use **mBERT** from the multilingual perspective. We initially used the Google Translate API to translate the dataset, but we got a timeout error due to the size of the dataset, so we divided the dataset into chunks of 5000 data points and then used the same Google Translate but via SpreadSheet, which didn't throw any errors. All three datasets are converted into Telugu, Tamil, and Hindi using the same technique. We have used the same parameters for the multilingual dataset as the English ones given by the Author.

1.4.2 Analyzing a Multilingual Model

We used the same method as mentioned above for converting test data into multiple languages so that there will be coherence in the way sentences are structured for train and tests. While converting datasets to a different language, we saved them individually, so when training the mBERT, we trained a model on each language for each percentage of training data. Which makes the number of experiments to 48. We stick to the same old metric Accuracy since this is a semi-supervised model.

1.5 Likely challenges and mitigations

There are two hardest things in this work, the first one is to hand annotate a huge amount of sentences with specific intents which could be biased based on the annotator. Also, the next big challenge is to make sure the semi-supervised model labels the data with more reliability and accuracy. If things were harder than expected, we can write a BERT model and train it iteratively with pseudo labeling using only top-k elements from each cluster to decide the label for unlabeled data.

2 Related Work

Pseudo Labeling: Many papers did the pseudo-labeling, where they directly used the model to acquire labels to the unlabeled data by just using the argmax from the model output. But this doesn't consider the confidence of the model while giving a pseudo label. The main issue with this method is that mislabeled data in the early iteration can degrade the overall performance of the system.

Another common practice to help reduce the mislabeled samples is to use a threshold τ to ensure that only high-quality labels are retained for the next iteration. The problem with this is that we will have very little data to infer upon each iteration of data. which is the main strategy of this paper(A. Gera and Slonim, 2022).

(E. Arazo and McGuinness, 2020) discussed the problem of the model becoming very biased towards the easy classes in the early iterations of learning. methods such as FlexMatch (B. Zhang, 2021a) have tried to solve this problem by allowing each class to have its threshold. However, this doesn't help since it will have a lower confidence threshold for the lesser-known classes in the data.

(R. Wang, 2022) uses a contrastive learning-enhanced nearest neighbor mechanism for multi-label text classification, which lays out the steps for the current paper to enhance this.

Distance-based Pseudo labeling: (Z. Zhu and Liu, 2022) proposed a training-free way to detect corrupted labels. The results of this work show that distance-based labeling can be done in the latent space representations also means that nearest elements in latent space can have the same/similar labels.

(F. Taherkhani and Nasrabadi, 2021) uses the above assumption with Wasserstein distance to do the pseudo-labeling via clustering.

finally (G. Ding and Porikli, 2019) proposed affinity-based pseudo-labeling that uses cosine similarity between the cluster centers and unlabeled examples. The pseudo label is given based on the highest similarity score calculated.

Finally, Our paper adds a top-k strategy on top of clustering and gives the labels to only those k elements in one iteration based on the score which is a combination of prediction probability and cosine similarity. We do this procedure iteratively to label all the data points.

3 Experiments

3.1 Datasets

A large set of datasets has been used, by the authors, for the sake of the experiment in this paper. The three datasets used are CLINC150, BANKING77, and HWU64 and these datasets are publicly available in the GitHub repository. The CLINC150 contains 150 in-domain intent classes from 10 different domains, and one is an out-of-domain class. This is designed in such a way as to evaluate intent classification models across various domains. BANKING77 is a dataset comprising 77 intents related to the banking domain, while HWU64 has 64 intents from 21 different domains. Then based on the experiment we only take a percentage of labeled data and then the remaining data as unlabeled for the semi-supervised task such as 1%, 2%, 5%, and 10%. While working on one of these datasets, the data is split into three sections for performing the training and testing of the model: the training set, the validation set, and the test set, where the values of each set can be seen in Table 1. The training set contains a percentage of labeled examples used to train the model, and test sets are used to evaluate the final performance of the trained model. The validation set is used to tune hyperparameters and monitor the performance of the model during training. For the datasets BANKING77 and HWU64, a validation set was not available publicly, and thus, the authors created their validation set from the training set and made it available on GitHub. A summary of the dataset is provided in Table 1.

Table 1: Summary of Dataset Information

Dataset	Intents	Domains	Train	Val	Test
CLINC150	151	10	15,250	3100	5550
Banking77	77	1	9002	1001	3080
Hwu64	64	21	8884	1076	1076

All converted datasets are now available on our GitHub repository.

3.2 Implementation

This link https://github.com/hem1999/CS678_Project_tk-knn.git is our GitHub repository where the source code and converted language-wise datasets are available, and it is hugely inspired by the Author's GitHub. with small changes in code and installations of new dependencies, we can run the code and produce the results from Author's work. Let's see the implementation details.

Overview: Initially we prepare a percentage of labeled data and the remaining as unlabeled data. Then we train a BERT model with the supervised data. After that, we use that model to generate the pseudo-labels for the unlabeled data. Then we compute a similarity score for each unlabeled data that is calculated based on cosine similarity and prediction probability concerning the labeled data. Then we rank the examples so that we only take top-k sentences from the pseudo-labeled set and move those as the labeled data. Now we have more labeled data for the next iteration. We do the cycle for a pre-defined time or with early stopping criteria to break this loop.

$$\text{score}(u_m) = (1-\beta) \times p_{\text{model}}(y|u_m;\theta) + \beta \times \text{sim}(z_n, z_m) \tag{1}$$

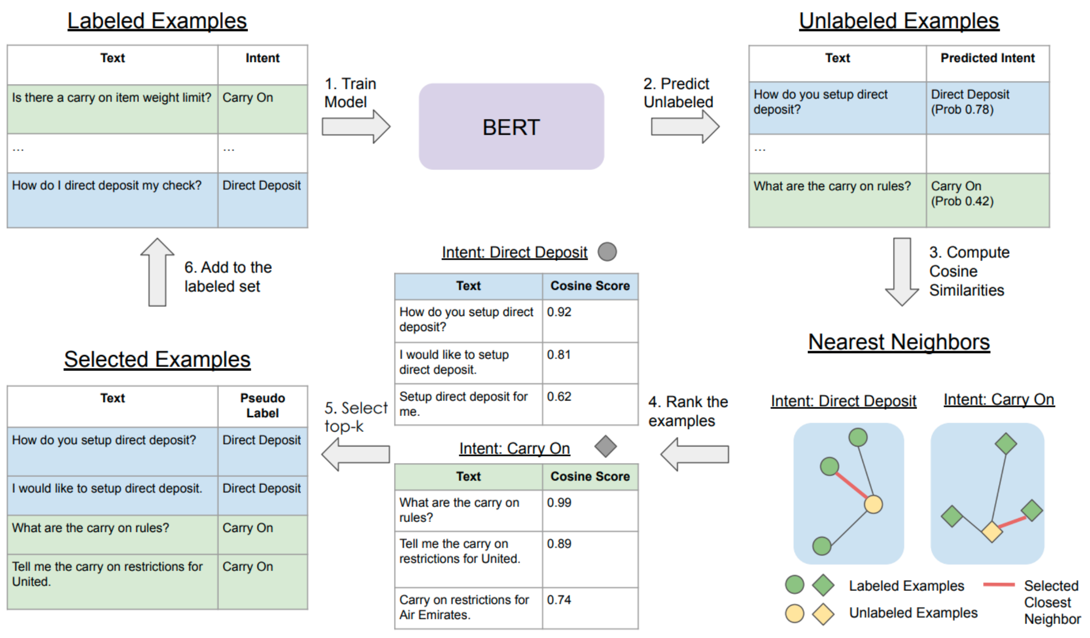
Loss Function: The Author has proposed a new loss function which is an augmentation of cross-entropy, supervised contrastive loss, and (Kozachenko and Leonenko, 1987) differential entropy estimator. The first cross-entropy loss is for the classification, then supervised contrastive (P. Khosla and Krishnan, 2020) is to ensure the model learns the good discriminative features in latent space. The final differential entropy is to make sure to spread the representations more uniformly using an extra parameter γ. Below are the equations:

Cross Entropy Loss:

$$L_{CE} = -\sum_{i=1}^{C} y_i \log(\hat{y}_i)$$

Supervised Contrastive Loss:

$L_{SCL} = \sum_{i\in I} \frac{-1}{||P(i)||} *$



Text	Intent
Is there a carry on item weight limit?	Carry On
...	...
How do I direct deposit my check?	Direct Deposit

Text	Predicted Intent
How do you setup direct deposit?	Direct Deposit (Prob 0.78)
...	
What are the carry on rules?	Carry On (Prob 0.42)

Text	Cosine Score
How do you setup direct deposit?	0.92
I would like to setup direct deposit.	0.81
Setup direct deposit for me.	0.62

Text	Cosine Score
What are the carry on rules?	0.99
Tell me the carry on restrictions for United.	0.89
Carry on restrictions for Air Emirates.	0.74

Text	Pseudo Label
How do you setup direct deposit?	Direct Deposit
I would like to setup direct deposit.	Direct Deposit
What are the carry on rules?	Carry On
Tell me the carry on restrictions for United.	Carry On

Figure 1: Overview of the Implementation process

$\sum_{p \in P(i)} \log \frac{\text{sim}(z_i, z_p)/\tau}{\sum_i a \in A(i) \text{sim}(z_i, z_a)/\tau}$

KoLeo's differential entropy estimator:

$$L_{KoLeo} = -\frac{1}{N}\sum_{i=1}^{N} \log(p_i)$$

where $p_i = \min_{i \neq j} ||f(x_i) - f(x_j)||$

Finally Author's loss function is:

$$Loss = L_{CE} + L_{SCL} + \gamma L_{KoLeo}$$

3.3 Results

As per the paper, their experiments did much better than the models than the existing models at the time. Even though we are not able to get the same accuracies as the author, our reproduced modelscano beat the accuracies of the existing models that are beaten by the author's paper. The below table comprises everything about our experiment results.

Table 2: CLINC150 Dataset results

Method	**1%**	**2%**	**5%**	**10%**
Author's CLINC150	53.73 ±1.72	65.87 ±1.18	74.31 ±0.96	79.45 ±1.01
reproduced CLINC150	48.42	59.82	69.47	74.63

Table 3: Banking77 Dataset results

Method	**1%**	**2%**	**5%**	**10%**
Author's Banking77	54.16 ±4.56	62.71 ±2.30	76.73 ±01.46	84.45 ±0.52
reproduced Banking77	50.86	59.12	72.66	77.89

Table 4: Hwu64 Dataset results

Method	**1%**	**2%**	**5%**	**10%**
Author's Hwu64	65.33 ±2.29	73.03 ±1.31	79.63 ±0.56	84.59 ±0.58
reproduced Hwu64	62.63	70.12	74.18	79.97

3.4 Checkpoint 2 Experimental Results

3.4.1 Multilingual Model Performance

With multiple languages, When we fix the percentage of the labeled data, among different languages there is some difference of +/- 10%. The below graphs show the experimental results of each dataset. For each dataset, we now have four versions, there are 3 language-specific datasets namely Telugu, Hindi, and Tamil and one more is a mixture of all languages including English.

Figure 2 shows the performance on the CLINC150 Dataset, where the mixed model got around 7% short by English for 10% labeled data. The same model gave excellent results for the individual language datasets with around +/- 5% of the mixed model. This tells that our model worked very well concerning individual language as well as the mixed.

Similarly, Figure 3 shows the performance on the HWU64 Dataset, the mixed model got around 10% less than the normal English model. While mixed models performed well, individual language models got a hit of another 10% accuracy. Which might be due to the high number of domains in the data.

For Figure 4, Again the accuracy of the mixed model is almost the same as the English model performance i.e.; 81% and 84% respectively. Individual language models performed similarly well on the Banking77 Dataset. These models got the best accuracy for the multilingual dataset since the data context is only "Banking/Financial Sector".

3.5 Discussion

Along with this paper, the authors made all the datasets, source code, and everything related to the paper available in the GitHub repository given by the author. While installing the virtual environment, we faced many issues, such as circular dependencies, some of which included required .dll files are missing, and a few libraries are missing.
The author has created well-structured code; he has a src folder that contains basic.py and ganBERT.py, where the implementations of Bert and basic neural networks reside. Then we have exp_configs.py, where we have all the configurations and inputs required for the experiment, such as the dataset, percentage of labeled data, the value of k, β, etc. Then we have trainval.py, where we are training the model, doing the cycles for self-training, early stopping, etc. There are no inconsistencies with the code.
We can reproduce the paper with a minor impact on accuracy, as mentioned in the results section, considering all the methods and datasets as seen in Table 2, Table 3, and Table 4. This is due to limitations in computing resources. The authors utilized a large setup of GPUs to obtain the results. During the final experiments and ablation studies, the TK-KNN model utilized approximately 4400 GPU hours. However, we were able to reproduce with comparatively lesser accuracy, comprising a total of 48 GPU hours. Considering the available resources, such as executing on ORC Hopper by George Mason University with limited GPU hours, this is the primary reason for the difference in reproduced accuracies compared to the ones ob-

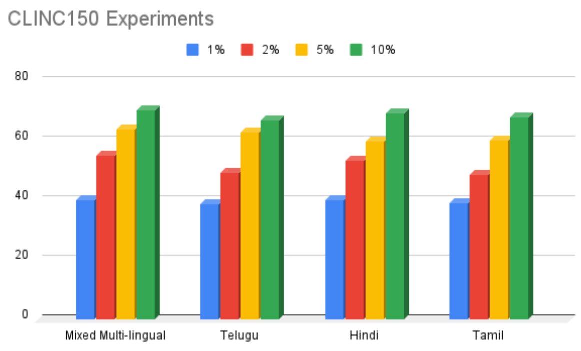


Figure 2: Graph showing how Mixed, Telugu, Hindi, and Tamil CLINC150 dataset performed on 1%, 2%,3%, and 5%

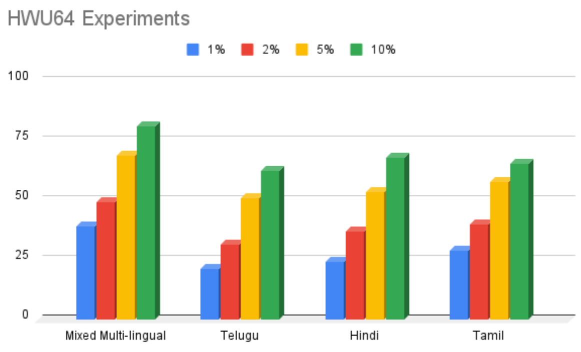


Figure 3: Graph showing how Mixed, Telugu, Hindi, and Tamil HWU64 dataset performed on 1%, 2%,3%, and 5%

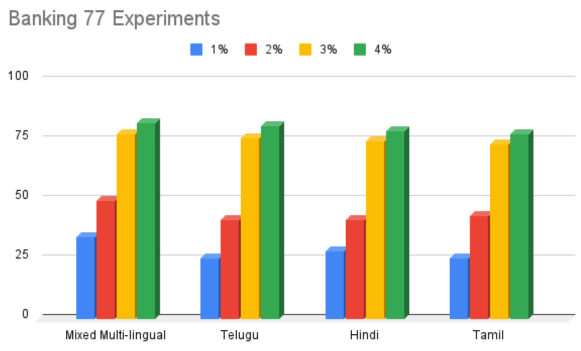


Figure 4: Graph showing how Mixed, Telugu, Hindi, and Tamil Banking77 dataset performed on 1%, 2%,3%, and 5%

Figure 5: Multilingual Dataset Performance

tained by the authors.

During the **second checkpoint**, one of the biggest challenges was translating datasets into multiple Indian languages like Telugu, Tamil, and Hindi. This required careful attention to language differences and accurate translations, which added more complexity to the model training process. The multilingual aspect of this project showed how crucial it is to make NLP models work well with various languages, something that isn't easy due to the availability of data and limited resources.

Given more time, we could have broadened the language coverage to include additional languages and experimented with a broader range of models beyond mBERT. Exploring alternative models could have provided better accuracy and robustness, as certain models may be better suited for specific languages or data characteristics. The multilingual nature of this work highlighted the need for adaptable frameworks capable of handling various languages efficiently.

From this experience, we learned that it's important to plan for resources because the amount of computing power required for each experiment is too high. So, plan for new architectures/models to solve it with less computing power. Also, researchers should focus on creating more flexible multilingual frameworks that work well with a wider range of languages. Lastly, to make NLP models more reliable and accurate, it's important to try different models and techniques beyond the ones we commonly use.

3.6 Resources

The author mentions that they have used almost 18,000 GPU hours for the experiments, For TK-KNN alone they have used 4,400 GPU hours. But we have used the hopper cluster for this, we did a total of 12 experiments each using 4 GPU hours on average which totals almost 48 GPU hours. Each GPU is A100 with 60GB Memory.

For **Check point 2**, for each of the 12 experiments done, we now have 4 sub-experiments spanning 3 language-specific and one mixed. This makes 48 Experiments and each requiring almost 2 GPU hours totals 96 GPU hours.

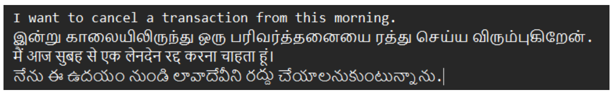
3.7 Error Analysis

One major observation is that the following sentences are labeled as "todo_list_update" but their original labels are as below:

(1) *I need a flight on Delta from Los Angeles to Seattle*, a label should be **book_flight** (2) *I want a new insurance plan*, the label should be **insurance_change** (3) *i have to cancel my reservation*, the label should be **cancel_reservation**
All of these are labeled as "todo_list_update", I feel it can be under that label if you are in the context of "todo app".

There are 150 intents overall in our dataset and there are intents that will clash with others based on the context. So I think these errors are due to a clash of intents in the training data which is true if the annotator thinking about the specific context. A similar confusion might happen to the model also.

3.7.1 Checkpoint 2 Error analysis



```
I want to cancel a transaction from this morning.
இன்று காலையிலிருந்து ஒரு பரிவர்த்தனையை ரத்து செய்ய விரும்புகிறேன்.
मैं आज सुबह से एक लेनदेन रद्द करना चाहता हूं।
నేను ఈ ఉదయం నుండి లావాదేవీని రద్దు చేయాలనుకుంటున్నాను.
```

Figure 6: Simple Example of small difference due to translations

There are translations, where that confuse even me due to words meaning different things in different contexts. The above sentences belong to the same English sentence but in different languages, after translation, it differs a little which could make the model wrongly assign labels. Along with the old reasons since the dataset has sentences that can be used under a different context which leads to different intent. Overall the reason could be that it means a little different meaning after translation or it is due to dependency on the annotator.

4 Plan for Checkpoint 2

Since the current work is about labeling a sentence with respective intent. The majority of users from a country like India use native languages rather than English which leaves space for us to work on multi-linguality. So, we would like to add more languages like Hindi, Telugu, Tamil, and Kannada.

The initial work of the author uses simple bert uncased with only English sentences. So, we are thinking of using Bert's multi-lingual model and having the same source code for other parts like top-k knn, cosine similarity, etc. The major remaining part is to have the data in other languages. One option we have is to use Google Translate to convert to Telugu, Tamil, Hindi, and Kannada. All of these languages are supported by the Google Translate API. Then we do the same set of experiments over each of the three available datasets using various percentages of labeled data

5 Workload Clarification

We as a team divided the tasks involved in checkpoint 1 among ourselves evenly. The first task was to pick the right paper from EMNLP 23 findings, where each one of us in the team took 30 papers, checking if the paper had well-documented source code or a publicly available repository. After 1.5 weeks of thorough evaluation, we narrowed our focus to the paper 'TK-KNN: A Balanced Distance-Based Pseudo Labeling Approach for Semi-Supervised Intent Classification.' This paper particularly stood out as it addressed a topic that resonated with all team members and was supported by comprehensive documentation. Each of us read the paper thoroughly and then we all together prepared a virtual environment on Hopper by resolving dependencies and minimal errors that occurred. Then we established that we have 3 datasets to perform experiments, which are taken by each of us. Then each performed experiments making sure that 1%,2%,5%, and 10% of the examples were labeled for the semi-supervised task. Then we together wrote the report. For Checkpoint 2, each team member took one dataset, contributing to its conversion and execution of the implementation and methodology. This collaborative effort ensured a thorough exploration of multilingual capabilities and comprehensive evaluation across diverse linguistic contexts.

6 Conclusion

The paper 'TK-KNN: A Balanced Distance-Based Pseudo Labeling Approach for Semi-Supervised Intent Classification' is supplemented by a comprehensive GitHub repository as referenced in the implementation section. This repository is well-documented, offering clear instructions for executing the source code and outlining all datasets used in the experiments. Our efforts to reproduce the study involved resolving several dependency issues within our virtual environment and making minor adjustments to the 'exp_configs' file to facilitate a broader range of experiments, initially not fully covered by the provided code. Despite these challenges and our limited computational resources, we successfully reproduced the authors' study. While our results did not fully match the accuracy reported by the authors due to fewer GPU hours and computing power, they still surpassed those of previous studies, demonstrating the robustness and effectiveness of the TK-KNN approach in the realm of semi-supervised intent classification.

For the **second checkpoint**, we extended the study's scope by translating all three datasets into Telugu, Tamil, and Hindi. We observed that while achieving multilingual support introduces challenges, the TK-KNN approach demonstrated robust performance across these additional languages. Achieving decent accuracies for all 48 models across three datasets highlighted the feasibility of adapting this semi-supervised intent classification model to multilingual contexts.

Potential future work involves expanding the language support further, enhancing the model's adaptability across more languages, and experimenting with other models and techniques to improve performance. A multilingual NLP framework built on these insights could better serve the global demand for diverse language processing in semi-supervised learning environments.

References

E. Shnarch Y. Perlitz L. Ein-Dor A. Gera, A. Halfon and N. Slonim. 2022. Zero-shot text classification with self-training. In *arXiv preprint arXiv*.

W. Hou H. Wu J. Wang-M. Okumura and T. Shinozaki B. Zhang, Y. Wang. 2021a. Flexmatch: Boosting semi-supervised learning with curriculum pseudo labeling. In *NeurIPS*.

P. Albert N. E. O'Connor E. Arazo, D. Ortego and K. McGuinness. 2020. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *IJCNN*.

S. Soleymani J. Dawson F. Taherkhani, A. Dabouei and N. M. Nasrabadi. 2021. Self-supervised wasserstein pseudo- labeling for semi-supervised image classification. In *CVPR*.

S. Khan Z. Tang J. Zhang G. Ding, S. Zhang and F. Porikli. 2019. Feature affinity-based pseudo labeling for semi-supervised person re-identification. In *IEEE Transactions on Multimedia, 21(11)*, page 2891–2902.

L. F. Kozachenko and N. N. Leonenko. 1987. Sample estimate of the entropy of a random vector. In *Problemy Peredachi Informatsii*.

C. Wang A. Sarna Y. Tian-P. Isola A. Maschinot C. Liu P. Khosla, P. Teterwak and D. Krishnan.

2020. Supervised contrastive learning. In *Advances in neural information process- ing systems*, page 18661–18673.

X. Dai R. Wang. 2022. Contrastive learning-enhanced nearest neighbor mechanism for multi-label text classification. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, page 672–679.

Z. Dong Z. Zhu and Y. Liu. 2022. Detecting corrupted labels without training a model to predict. In *ICML*.