**RVL-CDIP : Enhancing Few-shot Intent Classification with Transformer-based Models and Similar Datasets**

**Sumit Singh1 Pankaj Kumar Goyal2**

1. Indian Institute of Information Technology, Allahabad and sumitrsch@gmail.com
2. Indian Institute of Information Technology, Allahabad and pankajgoyal02003@gmail.com

**Abstract**

Intent classification, which involves recognizing the speaker's intention, stands as a crucial task in the realm of natural language understanding. This task involves intent classification with the fewer data hence it is a problem of Few-shot learning. We have provided 2248 samples for training and validation so there is also a challenge of proper validation data for the confirmation of best checkpoints during training. We have utilized similar datasets and transformer-based checkpoints, which are fine-tuned for intent classification. To deal with the lack of validation data, k-fold validation was applied for some cases. To leverage the advantage of the similar datasets, we also fine-tuned our task in two stages. We have also fine-tuned the previous existing dataset along with the hybrid approach in which the weights of previously finetuned checkpoints are frozen. Additionally, the Setfit model utilized for the classification task, which trained with some additional similar examples which are generated by Setfit. For some setups, augmentation is applied to generate new examples. Our best score (90.90 F1-score) was achieved by a majority voting ensemble over predictions of the best models.

All the codes and best model is available here:-

https://huggingface.co/pankaj10034/Indoml\_2023\_test

**Feature Selection/Method Description**

We have leveraged the advantage of transformer-based ( Vaswani, 2017) encoders like Roberta (,Liu, Yinhan, 2019) and xlm-Roberta (Conneau, Alexis, 2019) for feature selection. Our work performs multiple experiments for utilizing knowledge of pre-trained encoders. We have here basically trained our model using four methods which are the following:-

(i) Fine tuning in a single stage,

(ii) Fine-tuning in two stages

(iii) Hybrid model

(iv) Training using Set-fit trainer

Now let’s discuss all of these techniques in detail step by step:

**(1) Fine tuning in a single stage**:- It basically means that we are taking any pretrained model and fine-tuning it for our DownStream task which is basically the Intent Classification. Here for the fine-tuning, we have two options available either we can take any pretrained model that is already trained on a similar task, and we can also take Roberta large model because our task has all the data in the monolingual format, so Roberta large will be the best option if we are directly taking any sequence classification model that is not trained on the similar task. In both these cases, we have applied our k-fold cross-validation techniques by splitting the surprise dataset given into the 80% of the training set and 20% of the validation set and then taking five folds, so that we have a total of five different validation data available and our model can also be trained on the whole training dataset in the multiple folds.

Another technique we have applied here with both types of models is basically data augmentation because here we have around 2250 samples of the dataset available with around 150 classes, so for every class, we have a total of 15 samples available, which is maybe less for training and getting a good accuracy so in the data augmentation we have two options available which are Synonym replacement and random insertion. Let’s discuss both these Data Augmentation techniques and how we have applied data augmentation in our dataset so that we have a sufficient number of examples or datasets for training the model.

**(1) Synonym Replacement (** Dai and Adel. 2020 **):**- In this technique, we replace words in the text with synonyms while maintaining the original context. It's often used to diversify the vocabulary of a text, create more variation in the data, or make text more suitable for different audiences.

**Example**:

Original Sentence: "The quick brown fox jumps over the lazy dog."

Synonym Replacement: "The speedy brown fox leaps over the indolent dog."

In this example, synonyms like "quick" and "brown" are replaced with "speedy" and "leaps" while maintaining the original sentence's meaning.

**(2) Random insertion (** Dai and Adel. 2020 **)**:- Here we insert new words into our original sentence at random positions. We have to add words in such a way that the context or meaning of the sentence doesn’t change, otherwise, that sentence generated after random insertion may represent another intent instead of the intent of the original sentence and it may degrade the performance of the model.

**Example:**

Original Sentence: "The cat sat on the mat."

Random Insertion: "The agile cat unexpectedly sat on the comfortable mat."

In this case, the words "agile" and "unexpectedly" have been randomly inserted to create a longer and more complex sentence.

These Data augmentation techniques are used when we have to generate additional training data for the machine learning models.

Here we have very little data available as we have discussed above, so we have added both or one of the data augmentation into our data to make our data more robust and diverse and it will also increase the average number of samples per intent.

We have first applied the Synonym replacement where for every sentence we have generated 2 more sentences using this, so that now our dataset size becomes 3 times of the original dataset and then we have trained our model using the techniques we have discussed above.

Second thing here we have used is the combination of both the Synonym replacement and random insertion and generated 2 more examples for every sentence using both of these so that now our dataset size becomes the 5 times of original and after that we have followed the same process.

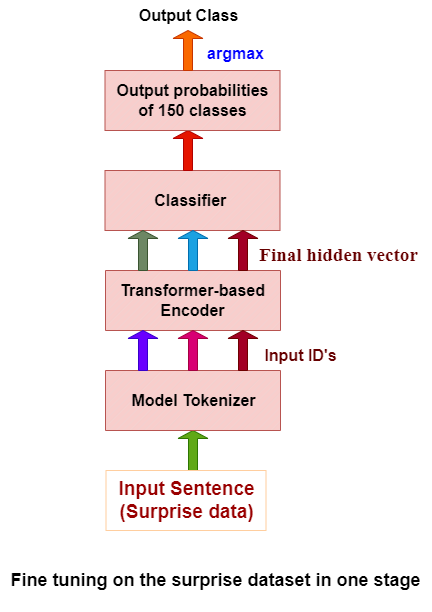


Fig.1 Architecture of Models Fine-tuned in a Single Stage

Here for the Fine-tuning in a single stage , we have first used Roberta-large pretrained model on the sequence classification task and fine tune that for our intent classification task by varying the learning rate in the range of 1e-5 to 7e-5 and batch size is either 32 or 64 and trained it on the 24GB RAM GPU. Other model parameters are default.

Next, we have taken the pre-trained models having good accuracy from the hugging face, which are trained for the same intent classification tasks and fine-tuned them for our Downstream task so that model can directly incorporate the pretrained knowledge with limited amount of data. While taking the model we have ensures that models is pretrained on the sufficient number of intents or labels we can say, because taking the model with tr￼

**(i) habana-xlm-r-large amazon massive:-** since in the development phase, we have given the massive multilingual dataset ( FitzGerald, Jack G. M.,2022) so we have first taken the xlm-roberta-large pretrained multilingual model which is trained on total 60 intents in the total of 51 languages and change the classification layer of the model and then fine tune it with a learning rate of around 2e-5 with the batch size of 32.

**(ii) ibm/roberta-large-vira-intents (**Gretz, Shai,2022**):-** Since the final phase consists of a monolingual English dataset and for the English dataset taking the Roberta large model if our dataset is less will be the best choice and in this case, we also don’t have the requirement of much of the resources, and we can train it easily on 12GB RAM of GPU, So we have used here IBM roberta-large pretrained model trained on the 180 intents which is more than this. Note that the model has been pre-trained with the batch size for both training and validation, and the learning rate is 5e-6, but for the fine-tuning tasks, we have taken the same parameters as the previous model. To get more results and the model, we can carry out more experiments but I have to keep it here limited.

**(iii) vira-chatbot/roberta-large-vira-intents-live:-** This model is pretrained on the 237 intents, and we have in the same way used for fine-tuning

Till now we have discussed all the pretrained models that we have fine-tuned for classifying our intents.

**(2) Fine-tuning in 2 stages:-** in this strategy, we have first found a similar intent dataset from the huggingface named “**xjlulu/ntu\_adl\_intent”** containing 15,000 training samples, 3,000 validation samples and 7,500 testing samples having a total of 150 intents exactly same as the intents of our surprise dataset. First, we have pretrained the Roberat large model on this dataset so that we can leverage the knowledge of this model while training our surprise dataset. Then, we selected the best checkpoint based on our validation score and used that for training our model on the surprise dataset(2248 samples) by splitting the dataset into 80% for training and the remaining 20% for the validation. After that, we have again selected the best checkpoint based on the validation score and used that for predicting the labels of our test dataset.

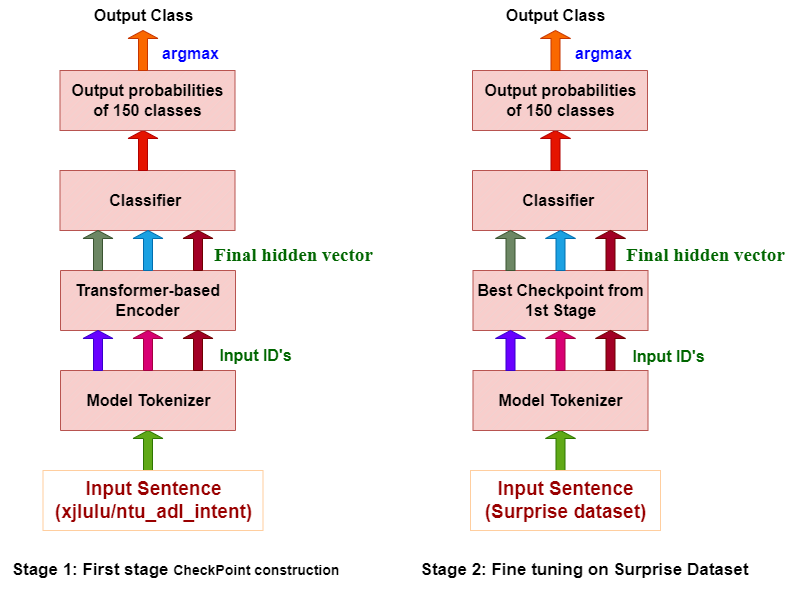


Fig.2 Architecture of Models Fine-tuned in two Stage

**(3) Hybrid model:** To take advantage of the previously fine-tuned model for the classification task we utilize some previously fine-tuned checkpoints and for training at surprise data, we freeze the checkpoints weights and add one extra encoder layer for the training so that checkpoint weights do not change and our model get enough weights for learning with the help of extra encoder layer and classification layer.

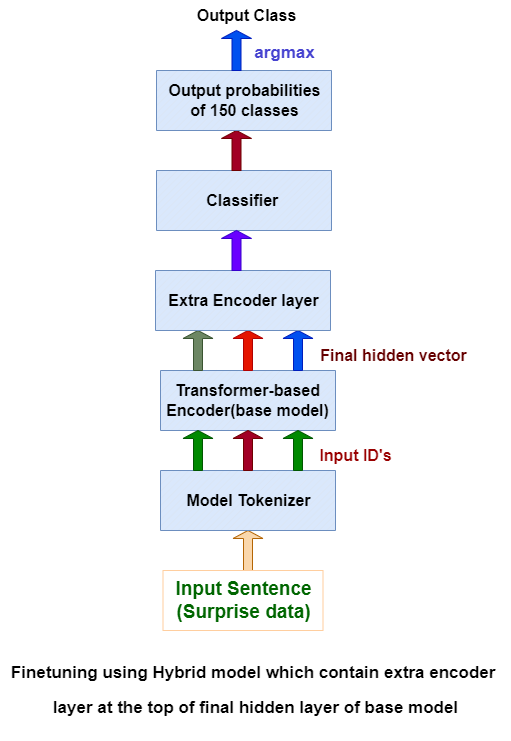
****

Fig.3 Architecture of Hybrid Model

**(4) Training using SETFIT** **trainer (**Tunstall, Lewis, 2022**):-**

Since the current techniques for few-shot fine-tuning require handcrafted prompts or verbalisers to convert examples into a format that's suitable for the underlying language model. SetFit dispenses with prompts altogether by generating rich embeddings directly from a small number of labeled text examples.

SETFIT (Sentence Transformer Fine Tuning) is an efficient and prompt-free framework for few-shot fine-tuning of Sentence Transformers (ST). SETFIT works by fine-tuning a pre-trained Sentence Transformers model on a small number of text pairs in a contrastive siamese manner. In the second step, a text classification head is trained using the encoded training data generated by the fine-tuned ST(Sentence transformers) from the first step. We have basically used two pre-trained models from the hugging face for training the model using the Setfit trainer, which are the following:-

**(i) sentence-transformers/paraphrase-mpnet-base-v2:-** This model is designed for paraphrase identification, which means it's trained to determine whether two sentences or phrases have the same or similar meaning and almost by the similar way we train the model using the Setfit, that’s why by default for intent related task or for the classification this model is used by the Setfit.

**(ii) ibm/roberta-large-vira-intents**

**(5) Voting ensemble:-** Since our work selected various checkpoints (which are already finetuned on divers intent classification datasets) for finetuning the our surprise data. Therefore our models, trained with different ways, have different knowledge for intent classification. Also with different random weights initialization at classification layer and different hyperparameters settings leads to different learning during experiment Therefore we ensemble the best fine tuned checkpoints based on accuracy at validation data.Majority voting method used for ensemble.We achieved our best score using the ensemble. This is a process of voting the prediction of best models.

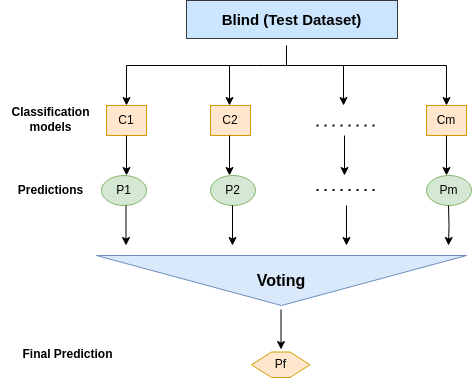


Fig. 4 Ensemble Architecture

**Experimental Results**

**Experimental Setup**

| Architecture | Model used | Learning rate(lr) | Batch size |
| --- | --- | --- | --- |
| Roberta | Roberta-large | 2e-5 | 32 |
| XLM-Roberta | Habana xlm-r | 1e-5 | 16 |
| Roberta | ibm/roberta-large-vira-intents | 4e-5 | 64 |
| Roberta | Vira-large | 1e-5 | 32 |
| Roberta | Roberta large 2-stage | 2e-5 (for both stages) | 64 |
| Roberta | Hybrid model | 6e-4(for extra layer) | 32 |
| Setfit | Setfit with Sentence transformers | 2e-5 | 16 |
| Setfit | Setfit with ibm/roberta-large-vira-intents | 2e-5 | 16 |

Table. 1 Hyperparameters of best models for each architecture.

Implementation of our task with the Cross entropy classification layer is done by xxxtokenclassification class defined in (Wolf et al., 2020), where xxx refers to the selected model. We trained the model with 1e-5, 2e-5, 3e-5, 4e-5 and 7e-5 learning rates and 16, 32 and 64 batch sizes. Hyperparameters for the best model are shown in Table 1. For optimization, AdamW optimizer was utilized. The cross-entropy loss Function with a linear layer converts the final features vector into class probabilities. All the implementations except Setfit and Hybrid are done by XXXForSequenceClassification library defined in (Wolf et al., 2020), here, XXX is Roberta and XLMRoberta in our case. Setfit model implemented with SetFitModel class. The hybrid model is implemented by putting a transformer encoder layer with a Cross-entropy classification layer at the top of the base model ( transformer encoder). We did this so that we could freeze the weights for the base model to preserve the knowledge of the base model during training, as these models were fine-tuned on different classes.

We have divided surprise data for training and validation in ratios of 80:20 and 90:10. Also K-fold validation method used during single stage training. Our best model was selected based on validation loss at each epoch for each setup.

**Results and Analysis**

Our best score was achieved with the voting Ensemble method of the best models, and our individual

The best score was achieved with the 2-stage setup with the Roberta-large model.

Details about both the setups are defined in Table 2. Results with other setups are given in Table. 2 and details are provided next.

1. **Results with the Roberta architecture in one stage:** We fine-tune the different checkpoints, which are already fine-tuned for the intent classification task for transferring their knowledge to our Surprise data. We achieved 88.19, 87.05 and 87.41 F1-Score with Roberta-large, ibm/roberta-large-vira-intents with vira-chatbot/roberta-large-vira-intents-live respectively. Results with the Roberta-large checkpoint are better. It might be possible as the intent of other checkpoints (which are already fine-tuned for intent classification) are not the same.
2. **Results with the XLM-Roberta architecture in one stage:** We fine-tune the habana-xlm-r-large, which are already fine-tuned for the intent classification task for transferring their knowledge to our Surprise data. We achieved 87.27 F1-Score.
3. **Results with the two-stage fine-tuning with Roberta architecture:** In the first stage, we trained the model with “xjlulu/ntu\_adl\_intent” dataset and achieved an 84.59 F1-score. In the second stage, we fine-tuned the model at surprise data and achieved an 89.11 F1-score. This strategy works because the intents of both the data are the same.
4. **Result with Hybrid model:** To take advantage of the previously fine-tuned model for the classification task, we utilize some previously fine-tuned checkpoints using hybrid architecture but results do not improve and it achieved an 86.98 F1-score.
5. **Results with Setfit:** Since for the very less amount of surprise data it won’t be ideal in the initial phase to just take any pretrained model and fine-tune it for our task because it may sometimes lead to underfitting. In our case, we have first encoded our 150 labels, and the utterances we have kept as it is no need to get tokenize the utterances because Setfit do it by default, here in the SetfitTrainer it takes column\_mapping as one parameter where we pass the text and encoded labels. Another thing that we have to consider here is the number of epochs which have kept to 2. We can also take it in the range of 1 to 5, but taking the epochs above five won’t be ideal. Another parameter is batch size, which we have taken to 32. Users can take batch size to be any of 16, 32, or 64 based on their requirements. Next, we have num\_iterations, which basically means the number of examples to generate for every sentence based on minimizing the cosine similarity loss between the sentences. We have taken it to be 10, we can also try with 15 or 20, but taking more examples ( generating more examples) will increase the loss between the original sentence and the generated sentences, and it will affect the accuracy of the model.

We have tried with two models in which one of them is “**paraphrase-mpnet-base-v2”** and achieved an accuracy of 78.55% on the test dataset. It's particularly useful for tasks like text similarity, semantic matching, and paraphrase detection

Another model we have used is **“ ibm/roberta-large-vira-intents”** which is pretrained on the same intent classification task but with a different number of labels and gives an accuracy of 85.43%.

| Architecture | Model used | Accuracy | F1-Score |
| --- | --- | --- | --- |
| Roberta | Roberta-large | 88.41 | 88.19 |
| Roberta | ibm/roberta-large-vira-intents | 87.31 | 87.05 |
| Roberta | vira-chatbot/roberta-large-vira-intents-live | 87.51 | 87.41 |
| XLM-Roberta | habana-xlm-r-large | 87.45 | 87.27 |
| Roberta | Roberta large 2-stage | 85.18 in first stage and 89.36 in second stage. | 84.59 in the first stage and 89.11 in the second stage |
| Roberta | Hybrid model | 86.98 | 86.89 |
| Setfit | Setfit with Sentence transformers | 77.56 | 77.44 |
| Setfit | Setfit with ibm/roberta-large-vira-intents | 85.43 | 85.34 |
| Independent of architecture | Ensemble of best models | 91.27 | 90.90 |

Table. 2 Accuracy and F1-Score of best models for each architecture.

**Novelty**

For few-shot learning we have applied two-stage training as we got a similar and bigger dataset (“**xjlulu/ntu\_adl\_intent**) which have similar classes and we put the learning rate of second stage is less than first stage as in first stage we trained with bigger dataset so that knowledge of first stage can transfer to second stage smoothly.

**Conclusion**

Intent classification, a fundamental component of natural language understanding, presents unique challenges, when dealing with fewer data, making it a few-shot learning problem. With a dataset of 2248 samples for training and validation, the task is further complicated by the need for effective validation strategies. To address these challenges, we employed transformer-based models fine-tuned for intent classification and explored various techniques. K-fold validation was used to mitigate the scarcity of validation data, and a two-stage fine-tuning approach leveraged the benefits of similar datasets. Moreover, we introduced a hybrid approach, freezing the weights of previously fine-tuned checkpoints. In addition to traditional fine-tuning, we incorporated the Setfit model, which included synthetic examples generated by Setfit and applied data augmentation in certain setups. Ultimately, our efforts culminated in achieving a remarkable 90.90 F1-score, which was accomplished through a majority voting ensemble over predictions generated by our best models.

**References**

Conneau, Alexis, et al. "Unsupervised cross-lingual representation learning at scale." *arXiv preprint arXiv:1911.02116* (2019).

FitzGerald, Jack G. M. et al. “MASSIVE: A 1M-Example Multilingual Natural Language Understanding Dataset with 51 Typologically-Diverse Languages.” ArXiv abs/2204.08582 (2022): n. Pag.

Gretz, Shai, et al. "Benchmark data and evaluation framework for intent discovery around COVID-19 vaccine hesitancy." *arXiv preprint arXiv:2205.11966* (2022).

Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." *arXiv preprint arXiv:1907.11692* (2019).

Tunstall, Lewis, et al. "Efficient few-shot learning without prompts." arXiv preprint arXiv:2209.11055 (2022).

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Huggingface’s transformers: State-of-the-art natural language processing.

Xiang Dai and Heike Adel. 2020. An Analysis of Simple Data Augmentation for Named Entity Recognition. In Proceedings of the 28th International Conference on Computational Linguistics, pages 3861–3867, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Reimers, Nils, and Iryna Gurevych. "Sentence-bert: Sentence embeddings using siamese bert-networks." arXiv preprint arXiv:1908.10084 (2019).