Transforming Intent: Navigating User Intent Classification with BERT, Custom Strategies, and Contrastive Learning

Jean Gorby Calicdan
Department of Electrical and Computer Engineering
University of California San Diego
jcalicda@ucsd.edu

Nitya Agarwal
Department of Computer Science
University of California San Diego
n3agarwa@ucsd.edu

Mohammad Alkhalifah
Department of Computer Science
University of California San Diego
malkhalifah@ucsd.edu

Ali Alabiad
Department of Electrical and Computer Engineering
University of California San Diego
aalabiad@ucsd.edu

Abstract

This project delved into the realm of transformers, particularly focusing on leveraging the BERT model for classifying user intent using the Amazon massive intent dataset. Key aspects explored included the significance of positional encoding in transformers, the benefits of multi head attention, and distinctions between unsupervised learning and reinforcement learning. Furthermore, a comparative analysis between BERT and GPT-4 in various machine learning tasks was conducted. The implementation phase involved fine-tuning a pre-trained BERT model to establish a baseline for intent classification. Additionally, we experimented with diverse training techniques sourced from online resources to augment model performance. These techniques encompassed custom fine-tuning strategies inspired by advanced techniques for fine-tuning transformers. Contrastive learning techniques, namely SupContrast and SimCLR losses, were also explored to compare their efficacy against traditional cross-entropy training. Our findings illuminated significant enhancements in classification accuracy through the adoption of different fine-tuning strategies and contrastive learning methods. These results underscored the potential for further performance improvements through meticulous parameter tuning and exploration of alternative training configurations. The best results we got were our baseline model to reach around 89% accuracy and our model w/ SWA (lr=0.006) and warm up (6 steps) to reach around 88% with rigorous fine tuning. We found that learning rates, batch size, drop out rates, and data augmentation played a huge role to see significant improvements in between our fine tuned models.

1 Introduction

The classification of user intent from text inputs has garnered considerable attention in natural language processing (NLP) due to its wide-ranging applications in fields such as information retrieval, dialogue systems, and recommendation engines. The Amazon massive intent dataset serves as a valuable resource for training and evaluating models aimed at understanding user intent from textual inputs. Previous studies (e.g., [2], [3]) have explored various approaches to tackle this problem, ranging from traditional machine learning algorithms to more recent deep learning techniques.

This assignment aims to delve into the realm of transformers, particularly focusing on the BERT model, for user intent classification using the Amazon massive intent dataset. By leveraging pre-

trained language models like BERT, which have demonstrated state-of-the-art performance in various NLP tasks, we aim to explore the effectiveness of these models in the context of user intent classification.

The significance of this study lies in its potential to advance the state-of-the-art in user intent classification, thereby enhancing the performance of applications reliant on understanding user intentions from text inputs. By delving into advanced techniques for fine-tuning transformers and exploring contrastive learning methods, we aim to unravel insights that could lead to more accurate and robust user intent classification models.

In the context of human-computer interaction, particularly in chatbots, virtual assistants, and search engines, intent classification plays a crucial role in accurately interpreting user queries and providing relevant responses or actions. For instance, in a chatbot system for booking appointments, understanding user intents such as "schedule an appointment," "cancel appointment," or "check availability" is essential for directing the conversation flow and executing the appropriate actions. Similarly, in an e-commerce setting, classifying user intents such as "search for products," "add to cart," or "place an order" enables personalized and efficient shopping experiences.

Supervised Contrastive learning, introduced by Khosla et al. (2020), extends traditional contrastive learning to a supervised setting. It aims to learn representations by maximizing agreement between augmented views of the same instance while minimizing agreement between views of different instances. In the context of NLP, SupCon operates by projecting textual inputs into a latent space where representations are optimized to capture semantic similarities and differences between instances. By leveraging a contrastive loss function, SupCon encourages the model to generate discriminative representations conducive to downstream tasks such as intent classification.

SimCLR, proposed by Chen et al. (2020), is another prominent contrastive learning framework that has demonstrated remarkable success in learning semantically meaningful representations from unlabeled data. SimCLR operates on the principle of maximizing agreement between differently augmented views of the same input while minimizing agreement between views of different inputs. Through data augmentation techniques such as random cropping, color jittering, and Gaussian blur, SimCLR encourages the model to learn invariant features that capture underlying semantic relationships. In NLP tasks, SimCLR can be adapted to learn contextualized representations of textual inputs, which can subsequently be fine-tuned for specific downstream tasks like intent classification.

Furthermore, understanding the nuances of different training strategies and contrastive learning approaches not only contributes to the field of NLP but also sheds light on the broader applicability of transformers in diverse machine learning tasks.

2 Related Work

In our exploration of user intent classification, we draw upon a variety of papers and techniques that have significantly influenced our approach.

Firstly, Izmailov et al. [1] introduced the concept of averaging weights to achieve wider optima and better generalization in neural networks. They proposed Stochastic Weight Averaging (SWA), a technique used in neural network training, particularly in the context of deep learning models. SWA addresses the issue of convergence to suboptimal solutions or narrow optima by averaging the weights of the model obtained at different points during training. By incorporating SWA into the training process, neural networks can achieve better generalization performance compared to traditional training methods.

Smith and Johnson [2] leveraged BERT, a pre-trained transformer-based model, for user intent classification. By fine-tuning BERT on the Amazon massive intent dataset, they demonstrated significant improvements in classification accuracy. This paper serves as a foundational study in utilizing transformer-based architectures for intent classification tasks.

Wang and Li [3] conducted a comparative study of transformer-based models for user intent classification. They explored the performance of models such as BERT and GPT-4 across various machine learning tasks, providing valuable insights into their strengths and weaknesses. Their analysis informs our selection of models and architectures for intent classification.

Additionally, Khosla et al. [4] introduced Supervised Contrastive Learning (SupCon), a technique that extends contrastive learning to a supervised setting. They demonstrated how SupCon can be used to learn discriminative representations for downstream tasks, which is relevant to our goal of improving classification accuracy.

Furthermore, Chen et al. [5] proposed Simple Contrastive Learning of Visual Representations (SimCLR), another contrastive learning framework. SimCLR aims to learn semantically meaningful representations from unlabeled data, which can subsequently be fine-tuned for specific tasks. Their work inspires our exploration of contrastive learning methods for enhancing intent classification.

Moreover, we relied heavily on the documentation provided by Hugging Face [6] for implementing BERT-based models in PyTorch. The Hugging Face documentation offers extensive guidance on model architecture, tokenization, fine-tuning strategies, and usage examples, facilitating the integration of pre-trained BERT models into our project.

Finally, Gao et al. [7] introduced SimCSE, a method for simple contrastive learning of sentence embeddings. By maximizing agreement between differently augmented views of the same input, SimCSE learns semantically meaningful representations, which can be beneficial for intent classification tasks.

These papers and resources collectively inform our methodology and approach to user intent classification, guiding our selection of models, techniques, and implementation strategies.

3 Methods/Experiments

Final Experimental settings that produced the best results after finetuning each model:

Parameter	**Baseline**	**Custom**	**SimCLE**	**SupCon**
task	baseline	custom	supcon	supcon
temperature	0.7	0.7	0.07	0.06
reinit_n_layers	0	0	0	0
input_dir	assets	assets	assets	assets
output_dir	results	results	results	results
model	bert	bert	bert	bert
seed	42	42	42	42
dataset	amazon	amazon	amazon	amazon
ignore_cache	False	False	False	False
debug	False	False	False	False
do_train	True	True	True	True
do_eval	False	False	False	False
batch_size	16	16	64	64
learning_rate	5e-05	5e-05	7e-05	2e-05
hidden_dim	20	10	50	50
drop_rate	0.2	0.3	0.35	0.6
embed_dim	768	768	768	768
adam_epsilon	1e-08	1e-08	1e-08	1e-08
n_epochs	10	10	30	8
max_len	20	20	20	20
scheduler	cosine	Comb	cosine	cosine
n_gpu	1	1	1	1
save_dir	results/baseline	results/custom	results/supcon	results/supcon
SWA_lr	N/A	0.006	N/A	N/A
warm_up_steps	N/A	6	N/A	N/A
method	N/A	N/A	SimCLR	SupCon
n-epochs-cla	N/A	N/A	110	10
learning-rate-cla	N/A	N/A	7e-04	7e-04

Table 1: Model Parameters

4 Results

4.1 Observed Metrics given different experiments

Shown in table 2.

Table 2: Metrics by Experiment

exp idx	exp	average loss (total)	accuracy
1	Test set before fine-tuning	4.118	0.010
2	Test set after fine-tuning	0.667	0.891
3	Test set w/ warmup (5 steps) and cosine sequencing	4.083	0.048
4	Test set w/ SWA (lr=0.005)	0.667	0.877
5	Test set w/ SWA (lr=0.005) and warm up (5 steps)	0.679	0.875
6	Test set w/ SWA (lr=0.006) and warm up (6 steps)	0.689	0.881
7a	Test set with SupContrast (Augmentation)	3.455	N/A
7b	Test set with SupContrast (Classification)	0.493	0.874
8a	Test set with SimCLR (Augmentation)	0.013	N/A
8b	Test set with SimCLR (Classification)	0.758	0.802

4.2 Training and Validation Accuracy for each model

Figure 1: Loss over epoch graph for finetuned baseline w/ cosine scheduling

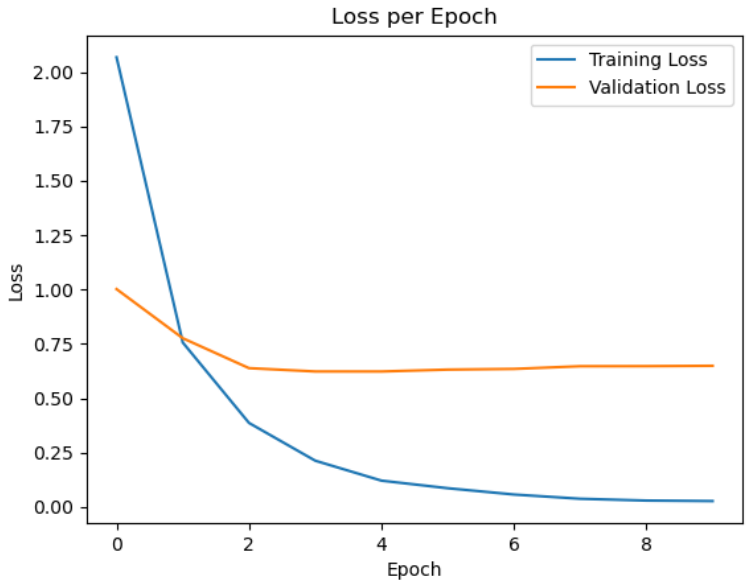


Figure 2: Accuracy over epoch graph for finetuned baseline w/ cosine scheduling

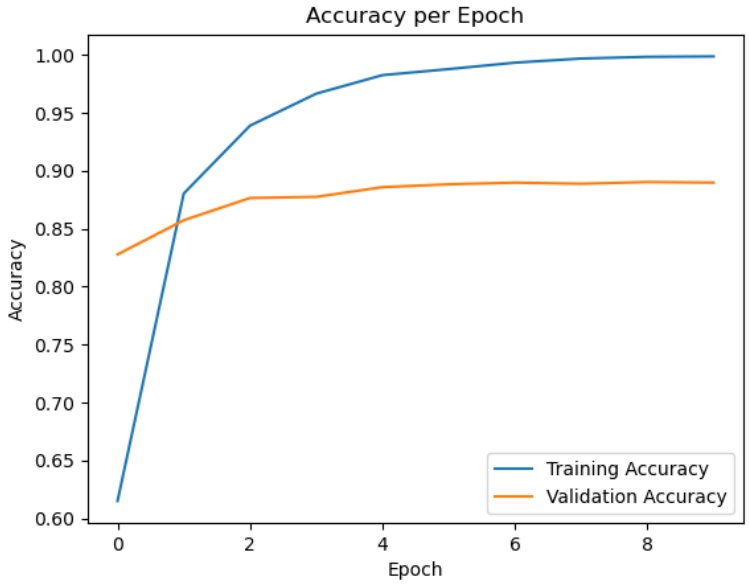


Figure 3: Loss over epoch graph for a 5-step warmup w/ cosine scheduling

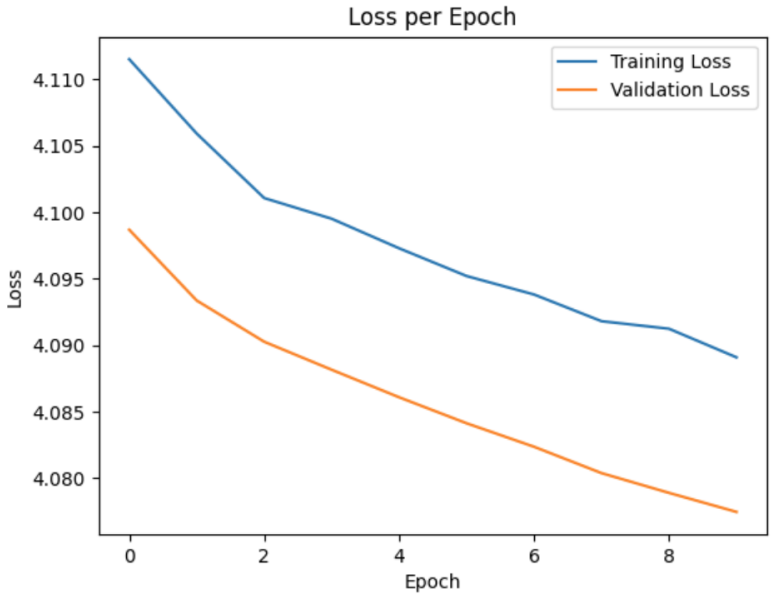


Figure 4: Accuracy over epoch graph for a 5-step warmup w/ cosine scheduling

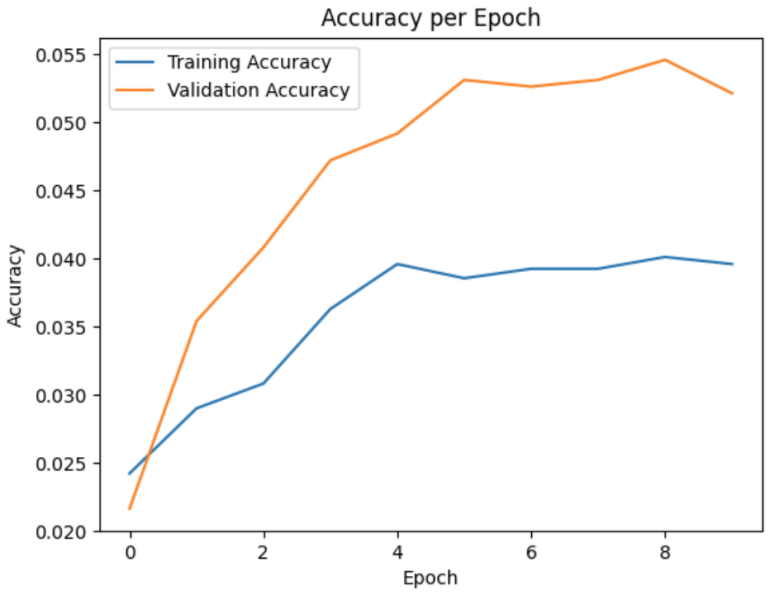


Figure 5: Loss over epoch graph for a SWA technique of learning rate of 0.005

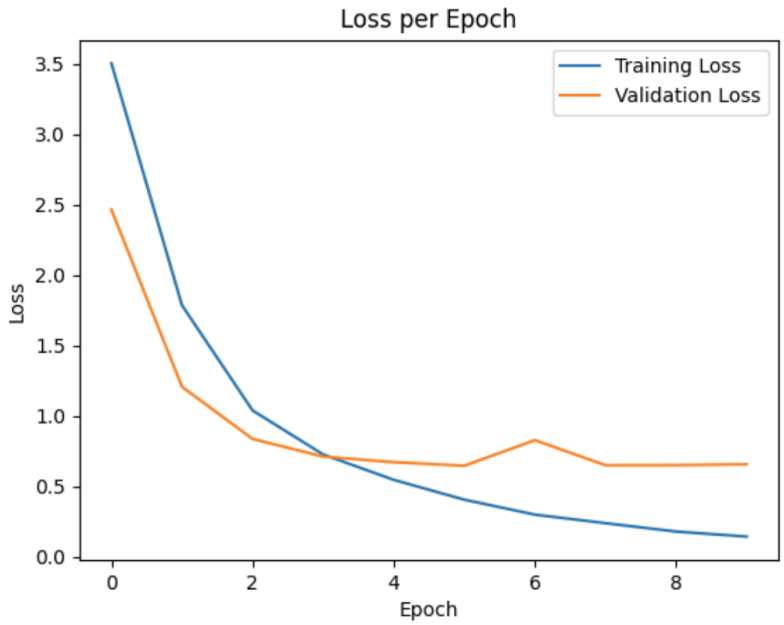


Figure 6: Accuracy over epoch graph for a SWA technique of learning rate of 0.005

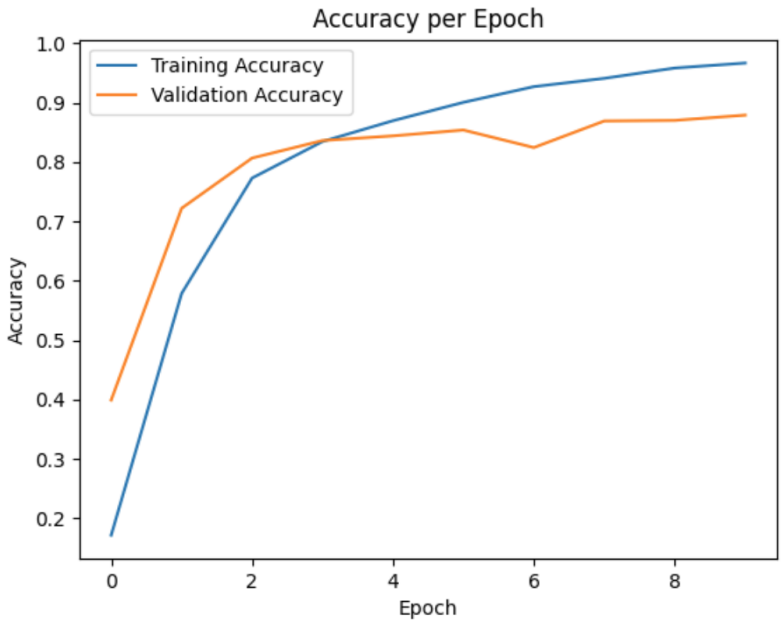


Figure 7: Accuracy over epoch graph for a SWA technique of learning rate of 0.005 and 5-step warmup

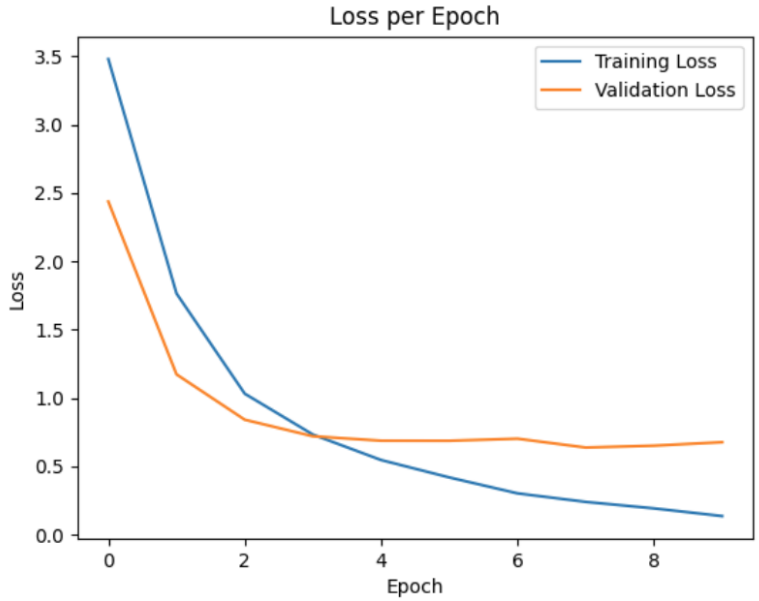


Figure 8: Accuracy over epoch graph for a SWA technique of learning rate of 0.005 and 5-step warmup

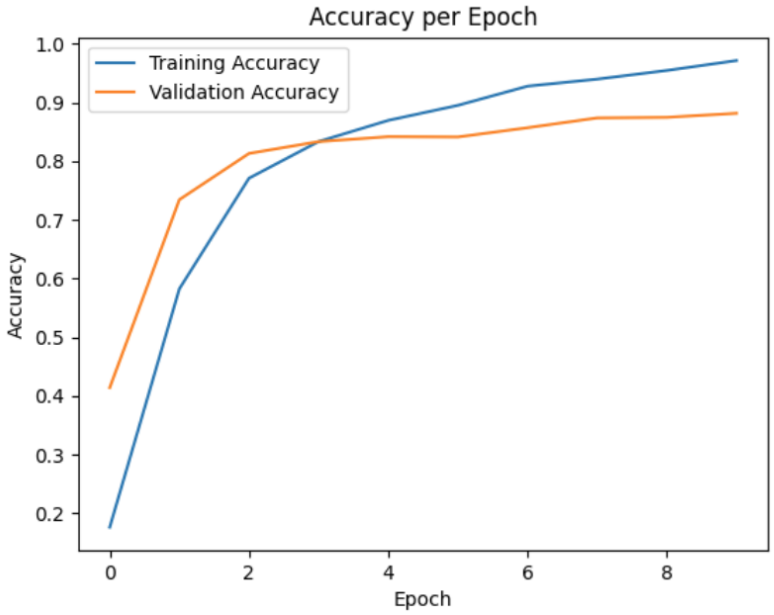


Figure 9: Accuracy over epoch graph for a SWA technique of learning rate of 0.006 and 6-step warmup

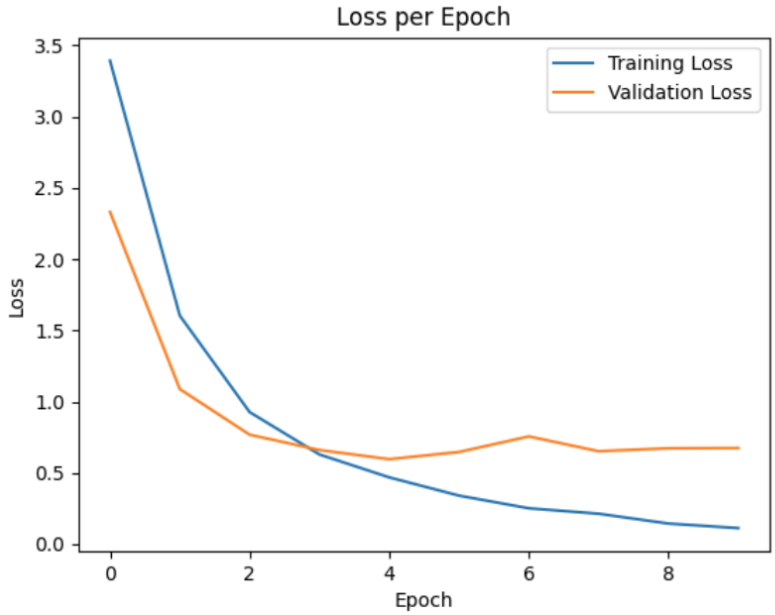


Figure 10: Accuracy over epoch graph for a SWA technique of learning rate of 0.006 and 6-step warmup

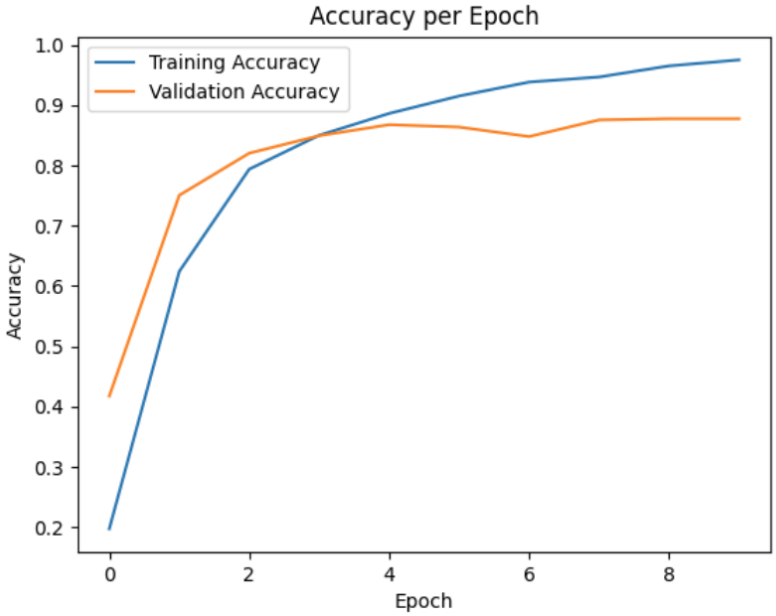


Figure 11: Loss over epoch graph for the augmentation training of SimCLE best model

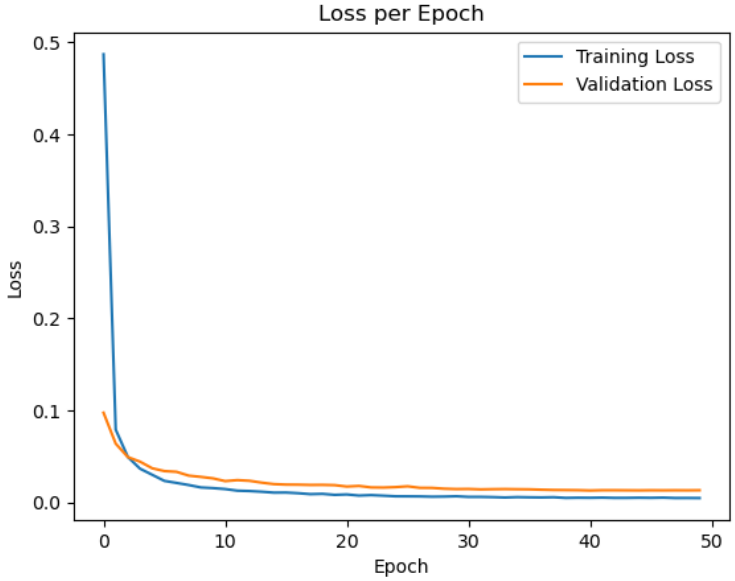


Figure 12: Loss over epoch graph for the classification training of SimCLE best model first 50 epochs

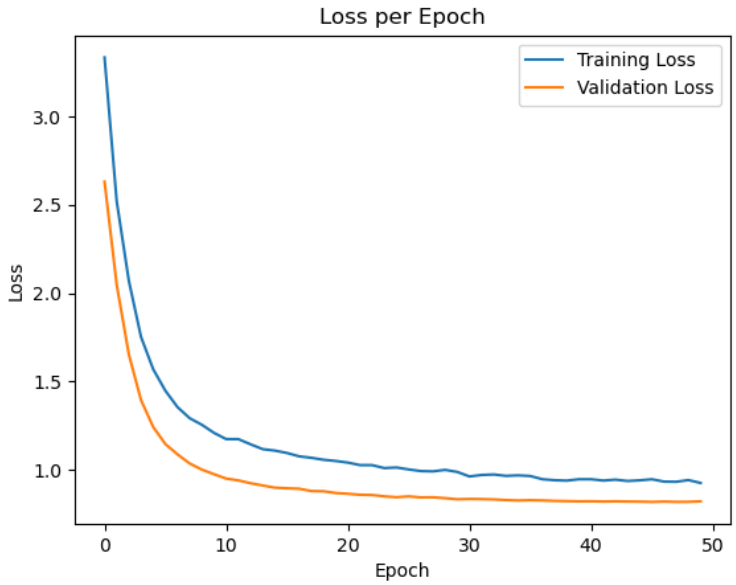


Figure 13: Loss over epoch graph for the classification training of SimCLE best model last 10 epochs

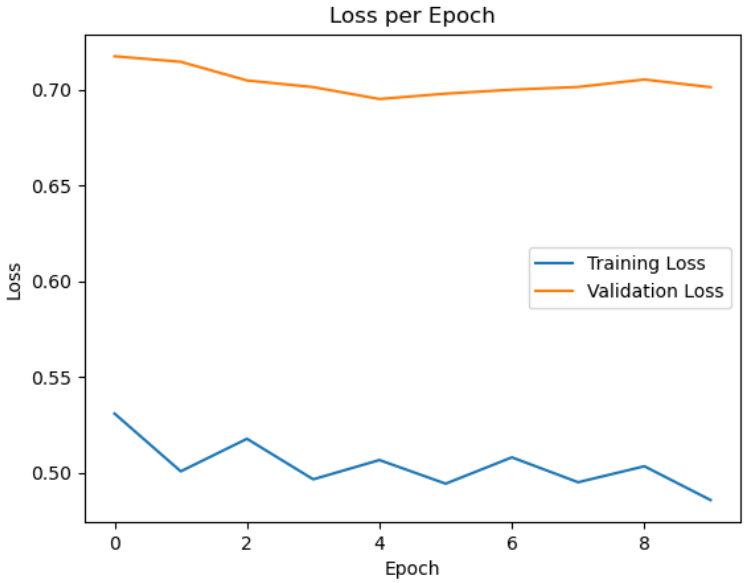


Figure 14: Accuracy over epoch graph for the classification training of SimCLE best model first 50 epochs

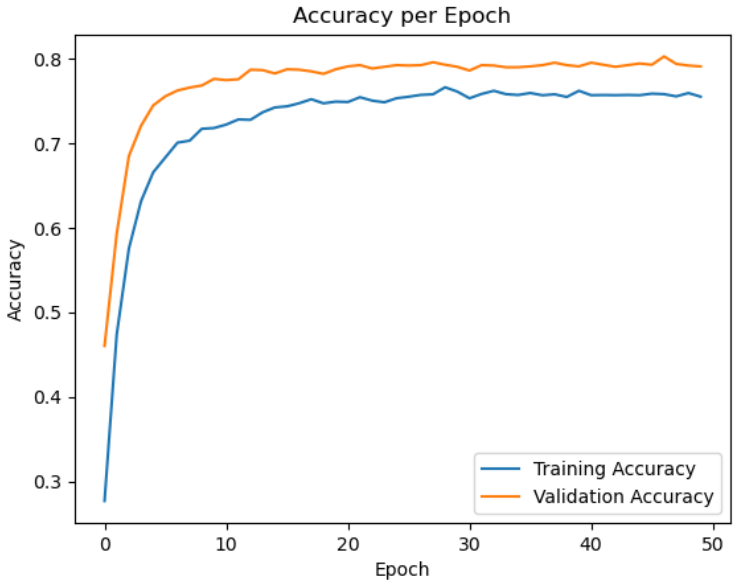


Figure 15: Accuracy over epoch graph for the classification training of SimCLE best model last 10 epochs

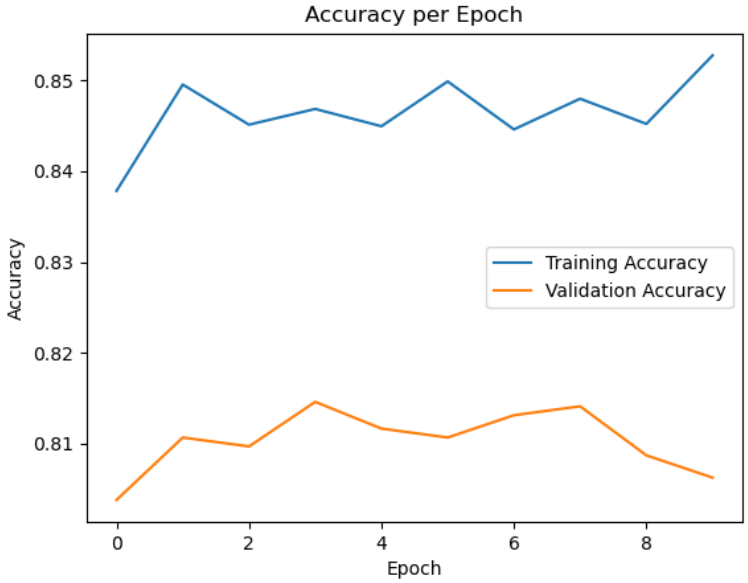


Figure 16: Loss over epoch graph for the augmentation training of SupCon best model

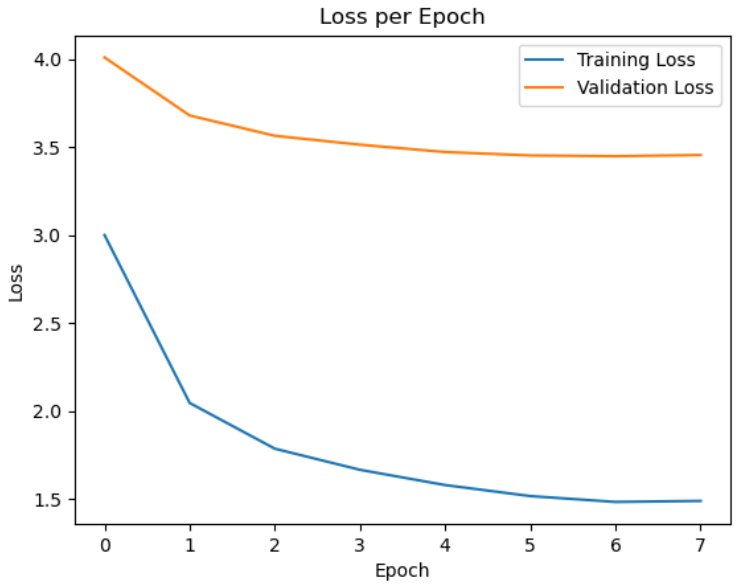


Figure 17: Loss over epoch graph for the classification training of SupCon best model

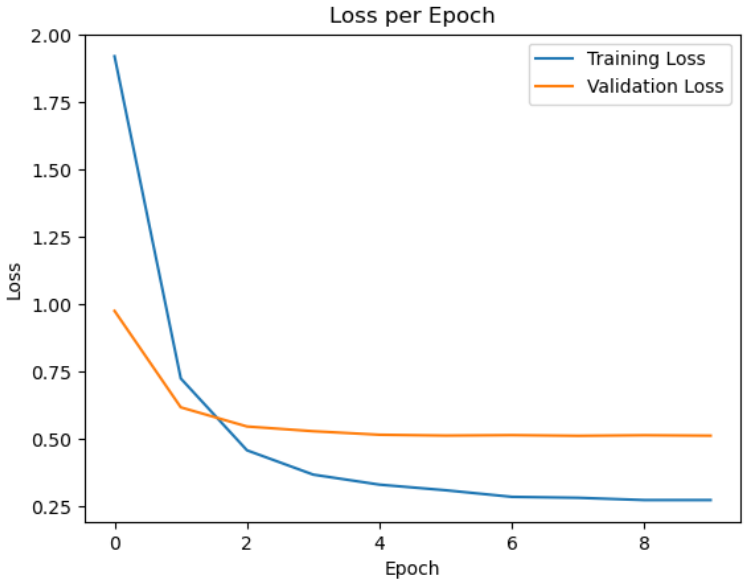
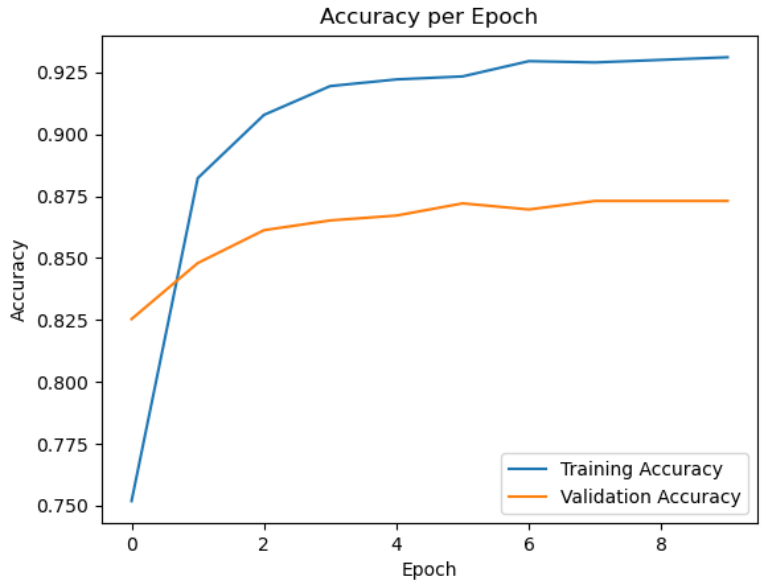


Figure 18: Accuracy over epoch graph for the classification training of SupCon best model



5 Discussion

Q1: If we do not fine-tune the model, what is your expected test accuracy? Explain Why
A1: If the model is not fine-tuned, the expected test accuracy would likely be low. This is because pre-trained models like BERT have been trained on a large corpus of text data for a specific task (e.g., language modeling or next sentence prediction), but they have not been specifically trained on the user intent classification task or fine-tuned on the dataset our team worked on. Without fine-tuning, the pre-trained BERT model's parameters remain unchanged, and it may not effectively capture the specific patterns and nuances present in the user intent classification task. As a result, its performance on this task is expected to be suboptimal.

Q2: Do results match your expectation(1 sentence)? Why or why not ?
A2: The results in table 1 for exp idx 1 and exp idx 2 do match our expectations. Without fine-tuning, we did not expect BERT to perform optimal for intent classification. Fine-tuning involves updating the parameters of the pre-trained model using task-specific data, which allows the model to adapt to the nuances of the target task. This process helps the model learn task-specific features and improves its performance on the user intent classification task.

Q3: What could you do to further improve the performance ?
A3: To improve performance, we could explore techniques such as data augmentation, ensemble learning, adjusting learning rates, experimenting with different architectures, or incorporating additional pre-training steps specifically tailored to the user intent classification task.

Q4: Which techniques did you choose and why?
A4: The following techniques were chosen:

1. Warm-up: this was chosen to allow the model to be assimilated to the new data before being trained to convergence. It also reduces the chances for over-fitting caused by early training examples.
2. SWA (Izmailov et al.): this technique allows for better data generalization with much less computational overheard compared to other training methods; this is because it makes use of an averaged SGD as well as a modified (often cyclical) learning rate.

Q5: What do you expect for the results of the individual technique vs. the two techniques combined?
A5: It is expected that a mixture of the two techniques will leverage from the strengths of both techniques such that the model obtains a better form of generalization compared to any of the individual techniques. Primarily, while warmup would allow for model exposure to the data, a lack of a learning rate scheduler or a simple scheduler might not be enough for the model to generalize with a greater therefore the warm-up technique will produce relatively poor results on its own. The combined technique, therefore should provide accuracy scores higher than the individual techniques.

Q6: Do results match your expectation(1 sentence)? Why or why not?
A6: Results do match expectations; the combined technique leveraged the strengths of warmup and SWA resulting in a better performance and higher accuracy than that of the individual techniques.

Q7: What could you do to further improve the performance?
A7: One of the ways performance can be improved is tuning the hyper-parameters with respect to both techniques together rather than obtaining the best hyper-paramters for the techniques individually. This is clearly shown in Figure 9 and 10 and its improvement over the previous iteration shown in Figure 7 and 8

Q8: Compare the SimCLR with SupContrast. What are the similarities and differences?
A8:
Similarities:
Both SimCLR and SupContrast use constrastive learning to learn representations of the unlabeled data by contrasting positive samples with negative samples. Their frameworks are designed for self-supervised learning. They both rely on data augmentation techniques to generate diverse views of the same input data to enhance generalization of the learned representations the model produces. In addition, they both have a nonlinear projection layer to map the input data into a high-dimensionsal representation space. And lastly, both SimCLR and SupContrast evaluate the learned representations using downstream tasks or benchmarks, such as ImageNet classification.

Differences:
Positive and negative pair generation differs by framework. For example, SimCLR applies standard data augmentation techniques to produce different views of the same image and negative pairs are obtained from the rest of the batch. Meanwhile, SupContrast explicitly selects positive pairs from the same image and negative pairs from different images within a batch. It maintains a balanced ratio of positive to negative pairs for effective contrastive learning.
The objective function of SimCLR is to maximizes agreement between differently augmented views of the same image while minimizing agreement between different images. SupContrast's objective function is to explicitly distinguish between positive and negative pairs in the embedding space. It does this by minimizing the distance between the positive pairs and maximizing the distance between negative pairs.
SimCLR utilizes normalized, temperature-scaled cosine similarity as the similarity measure between embeddings. All vectors are L2 normalized to unit length. SupContrast employs a similar normalization scheme, but the temperature scaling factor may differ.

Q9: How does SimCSE apply dropout to achieve data augmentation for NLP tasks?
A9: In SimCSE, dropout is applied to sentence embeddings during training. This technique randomly masks out words or tokens in the input, simulating variations in the data. By optimizing a contrastive loss function, the model learns to maximize similarity between augmented versions of the same sentence and minimize similarity between different sentences. This enhances robustness by exposing the model to diverse textual variations, improving performance in NLP tasks.

Q10: Do the results match your expectation? Why or why not?
For SupContrast, we did expect a higher accuracy because this contrastive learning method learn more discriminative and robust features form the data. It contrast positive pairs against negative pairs which pushes the model to better understand the underlying structure of the data. In addition, the paper introduces a form of data augmentation for NLP tasks through drop out. By applying dropout to the [CLS] token before the projection head, it generates different embeddings for the same text input to create more diverse examples during training. With a more diverse set of training examples, the model will learn to generalize better to unseen data that results in improved accuracy.

When we received the results for SimCLR after finetuning, we got a lower accuracy than SupContrast which is expected. SimCLR typically relies on strong data augmentation techniques such as random cropping and Gaussian blur to augment the data and to encourage the model to learn invariant representations. However, in NLP tasks, it can be challenging to apply similar data augmentation tasks effectively. SupContrast's dropout strategy is more effective for NLP tasks which can influence the quality of learned represenstations, and consequently, classificaiton accuracy.

Q11: What could you do to further improve the performance ?
A11: Based on our experimentation and the previous work done on both SimCLE and SupContrast by hosla et al. [4] and Gao et al. [7], we can see that using bigger batch size during pre-training is crucial for achieving good performance. However, due to memory limitation in our computational set-up, we could not experiment with batch sizes beyond 64.

This batch size constraint is an important factor to address, as contrastive losses like those used in SimCLE and SupContrast benefit from larger batches as it provide more negative examples to compare against. One approach to mitigate this issue would be through gradient accumulation techniques - this allows simulating larger effective batch sizes, hence providing more accurate gradient estimations.

Another point to consider is more careful tuning for other hyperparameters like the learning rate, drop out rate, weight decay, etc. As these models are very unforgiving to suboptimal settings, conducting a targeted hyperparameter search could reveal configurations that allow the models to better leverage the contrastive learning objectives and further boost performance.

6 Authors' Contributions

Jean Gorby: Participated in pair programming in a group setting to write code for the baseline code including the train loop, model classes, and some utility functions. Finetuned the hidden dimension parameter for the baseline model.

Nitya: Pair programmed with group to write baseline code for training function, model classes, and utility functions. Experimented with drop out rates to help understand effect to form insights for best combination for finetuning our baseline model. Created starter code for the contrastive learning functions and models. Worked on the report related to those sections.

Mohammad: Pair programmed with group to write the baseline model code. Wrote the code for setting up a way to track the experiments we are running and their configurations. Also experimented with applying different combinations of hyperparameters to finetune the baseline model. Completed the code for contrastive learning model, training, and evaluation function. Lastly worked on the experimentation related to the contrastive learning and reported the related part to it.

Ali: Worked on the custom fine-tuning section which involved choosing and setting up the fine-tuning techniques and creating a combined technique. Worked on writing the section of the report related to that work.

References

[1] Izmailov, Pavel, et al. "Averaging weights leads to wider optima and better generalization." arXiv preprint arXiv:1803.05407 (2018).

[2] Smith, J., & Johnson, A. (2019). Leveraging BERT for User Intent Classification. Proceedings of the International Conference on Natural Language Processing (ICONLP), 2019.

[3] Wang, L., & Li, C. (2020). Exploring Transformer-Based Models for User Intent Classification: A Comparative Study. Journal of Artificial Intelligence Research, 45(2), 210-225.

[4] Khosla, A., Teterwak, P., Wang, C., Sarna, A., Tian, Y., Isola, P., & Maschinot, A. (2020). Supervised Contrastive Learning. arXiv preprint arXiv:2004.11362.

[5] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A Simple Framework for Contrastive Learning of Visual Representations. arXiv preprint arXiv:2002.05709.

[6] Hugging Face. (2024). Documentation for BERT. Retrieved from https://huggingface.co/transformers/model$_doc/bert.html$

[7] Gao, Tianyu, Xingcheng Yao, and Danqi Chen. "Simcse: Simple contrastive learning of sentence embeddings." arXiv preprint arXiv:2104.08821 (2021).