Understanding the impact of class imbalance in BERT models for paraphrase identification

Team name: Un-Natural Language People
Authors: Anubhaw Mathur, Simon Tong, Alejandro Vasquez Quijada
CS 7643 - Deep Learning
Georgia Institute of Technology
{amathur41,stong8,avasquezq}@gatech.edu

***Abstract*— The goal of this project is to assess the impact of various methods to address class imbalance on the performance of a BERT model used for paraphrase identification using the Quora Question Pairs dataset. In particular, we will implement the following data augmentation strategies: (a) focal loss to weight down easy-to-classify pairs of sentences; (b) paraphrase generation using a pre-trained model (Parrot); (c) sample generation based on the transitive property of the dataset. We compared the different methods using the accuracy and precision metrics. We discover that some techniques do not appear to increase the precision or accuracy of the base model. The model that included data augmentation based on the transitivity property does appear to perform better than the baseline by around 4%.**

***Keywords*—** *data augmentation, BERT, paraphrase identification.*

I. Introduction

Quora is a user-maintained website that facilitates information discovery through questions and answers. The website was launched in 2009, and by 2020 it reached 300 million monthly visitors [1]. Given the user traffic, it is likely that some users could ask questions that have already been addressed. Therefore, it would be advantageous to automatically identify questions with similar aims but with different wording. As a result, it will be easier to find high-quality answers to questions in a faster and less expensive manner, resulting in a better experience for users and lower costs for Quora. The issue of finding similar questions was originally solved by Quora by manually labeling questions as duplicated and then implementing Random Forests Classifiers [2]. In 2017, Quora released the Quora Question Pair (QQP) dataset and started a Kaggle competition in an effort to enhance its predictions. Since then, this dataset has been used to conduct research on paraphrase identification (predicting whether two pieces of text are similar based on their intent) as well as other tasks such as paraphrase generation and question answering [14]. Other popular paraphrase identification datasets include the Paraphrase Adversaries from Word Scrambling (PAWS) [3], Paraphrase Identification Requiring Computer Science Domain Knowledge (PARADE) [4], and Microsoft Research Paraphrase Corpus (MRC) [5]. These datasets have been used to create benchmarks for comparing the performance of various methods.

The QQP dataset includes 404,302 pairs of questions, each with a binary label indicating whether the pair is duplicated. Non-duplicated pairs account for 255,027 (63%) instances, while duplicated pairs account for 149,263 (37%). Only three missing value cases are identified and removed from the dataset. In total, 537,933 distinct questions were identified, with 111,780 (21%) appearing more than once in the dataset and a maximum number of appearances of 157 for a single question. This dataset is not representative of the universe of questions asked in Quora, since multiple stages of a stratified sampling method were performed. Additionally, ground truth labels are inherently subjective, and the manual labeling process may be noisy because the true meaning of sentences can never be known with certainty. As a result, while the set of labels supplied by humans can be considered correct because it is based on consensus, it should not be considered perfect because some experts may disagree or incorrect labeling may exist [13]. Some sanitization measures were applied to the final dataset to improve the quality of the text, including the removal of questions with extremely long question details. Finally, the dataset has been de-identified and has no sensitive information about individuals or subpopulations. However, individuals can be identified by conducting a search of the questions on the website.

The top competitors in the 2017 Kaggle competition used a variety of models, including Decomposable Attention Neural Network [15], Enhanced Sequential Inference Model [16], Siamese LSTM [17], neural bag-of-words [18], and bilateral multi-perspective matching (BiMPM) model [19], among others. Since the publication of the Kaggle competition, new models have outperformed the ones implemented by the top competitors, including Few-Shot Learner (EFL), data2vec, Charformer-Tall [14], and Bidirectional Encoder Representation from Transformers (BERT), and its extensions like Decoding-enhanced BERT with Disentangled Attention (DeBERTA) [20], A Lite BERT (ALBERT) [21], Generalized Autoregressive Pretraining for Language Understanding (XLNet) [22], among others.

The foundation for this project will be a fine-tuned Bidirectional Encoder Representations from Transformers (BERT) model. This model has been used to detect text similarity on some of the mentioned paraphrase identification datasets [10, 11, 12]. BERT is currently one of the most advanced pre-trained language models, which incorporates the

bidirectional context of a sentence input via self-attention and learns general language representation, and it has outperformed previous language models and embedding methods for text classification tasks [9]. In this project, we will use the base BERT model (~110M parameters). Although other models such as RoBERTA, XLNET, and DeBERTA outperform BERT on some tasks by large margins (2-20% depending on the model), they also have long training times (between a 4X-5X increase in training time depending on the models) due to changes in the pre-processing steps [23]. On the other hand, simplified versions of BERT such as DistilBERT [24] and ALBERT have been developed. At the expense of some performance (3-10% depending on the model and the task), these models are up to 4X faster to train than BERT which can speed up inference speed [23]. Given the foregoing, we decided to use BERT as a baseline because of how well it manages the trade-offs between computational complexity and performance.

Finally, labeling duplicated questions manually takes time which may result in relatively small datasets to train models on, potentially creating a generalization problem for the predictions. We will estimate the effect of various data augmentation methods on the performance of the fine-tuned BERT model to solve this problem. We will specifically perform three data augmentation strategies: (a) resampling strategies [6], (b) paraphrase generation to create new samples using pre-trained models [7], and sample generation based on the transitive property of the dataset [8].

II. Methodology

A. BERT model

BERT is a natural language understanding model developed by Google in 2018 that is widely used in tasks such as sentiment analysis, text classification and named entity recognition. BERT was trained on massive amounts of training data including all the words in Wikipedia (~2.5B words) and Google's BooksCorpus (~800M words). The learning process for a text is bidirectional learning, which involves masking a word and enforcing the model to predict it based on the contextual words. This procedure is enhanced by Next Sentence Prediction (NSP), in which the model attempts to predict whether or not a given sentence follows another [25]. The architecture for BERT is based on transformers, which allow for parallelization of attention mechanisms, and result in a more efficient learning process. Nevertheless, BERT is only based on an encoder, which is a stack of layers that reads the input and generates a language representation.

The BERT architecture is depicted in Figure 1. The architecture is composed of a series of encoders that are made up of stacked layers that include multi-head attention, normalization and feed forward. In the paper that introduced BERT, the researchers trained two models: a base BERT (12 layers, 12 attention heads and 110M parameters) and a large BERT (24 layers, 16 attention heads and 340M parameters) [9]. In this project, we will use the base model and tune the hyper-parameters for the paraphrase identification task. To convert this architecture into a text classification model, the original BERT paper suggests adding extra linear layers with a softmax activation as the final layer. In the pre-processing step, we will join each pair of sentences with a separator between them and we will add a CLS token that represents the beginning of the input.

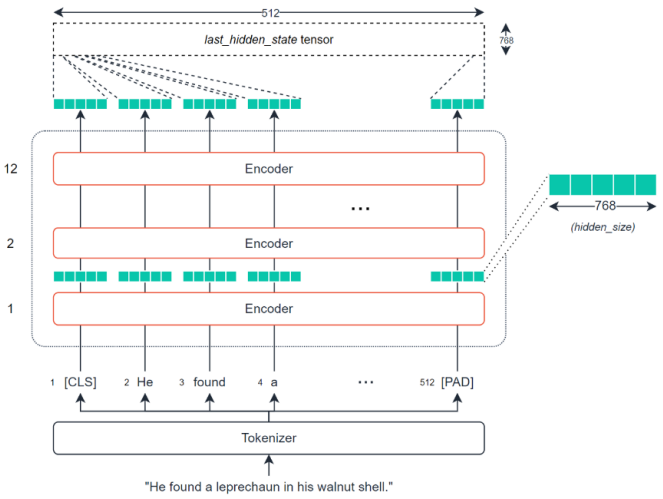

Figure 1. Base BERT architecture [9]

B. Data augmentation methods

1) Class imbalance methods

Class imbalance problems have been widely studied in the machine learning literature [32,33]. The class imbalance problem arises in supervised learning models where there is a large difference in sample sizes between two or more groups (minority and majority groups). The issue with class imbalance is that models will tend to over-classify the majority group, resulting in poor results when confronted with minority groups. There are three types of methods for dealing with class imbalance issues: (a) data-level methods that use different sampling methods to reduce imbalance; (b) algorithm level that commonly use weight or cost schemas or modify the models to reduce the bias; (c) hybrid systems that combine both. Some of the methods traditionally used in machine learning models include the Synthetic Minority Over-Sampling Technique (SMOTE), the Random Undersampling (RUS) and Random Oversampling (ROS), and other methods like DataBoost-IM [34]. However, the class imbalance problem has not been widely studied in the context of deep learning models [32]. Among the data-level approaches to deal with imbalanced datasets in deep learning are transfer-learning with data-augmentation [35], dynamic sampling methods [36], and two-phase learning with sampling methods, among others [37]. Algorithm-level methods, on the other hand, include novel loss functions [38,39], as well as techniques for cost-sensitive learning [40,41]. Finally, the hybrid approach includes methods for feature-oversampling, as well as novel sample mining functions [42,43].

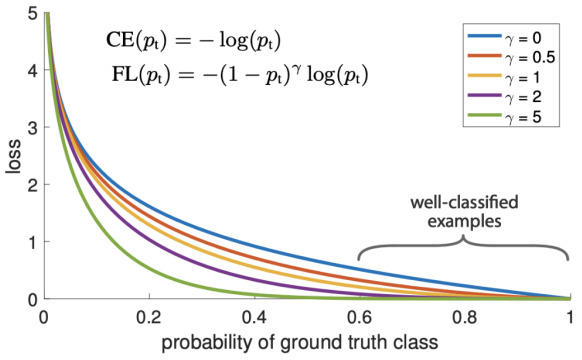

FIGURE 2. FOCAL LOSS FUNCTION [39]

In this report, we will use Focal Loss (FL) as an algorithm-level method to deal with the class imbalance in the BERT models. The Focal Loss (FL) is a classification loss function that is used to correct class imbalance during training [39]. Originally developed for object detection tasks, it has since been widely used to modulate the standard cross-entropy loss function in order to focus on learning difficult misclassified examples. This function dynamically scales down the contribution of easy-to-classify examples (majority groups) to rapidly improve predictions on hard examples (minority groups). Technically, the FL includes a factor term that multiplies the standard cross-entropy function. This factor allows to reduce the relative loss of well-classified samples while increasing it for poorly-classified samples. The function is shown in equation 1. This expression is commonly referred to as the alpha form of the FL. For simplicity, we assume that $\alpha = 1$, as originally defined in [39].

$$FL(p_t) = -\ \alpha(1 - p_t)^{\gamma} log(p_t) \quad (1)$$

where $\gamma > 0$, $\alpha = 1$, $p_t = I(y = 1)$, and $I(\cdot)$ is an indicator function.

2) PARROT paraphrase generation

Paraphrase generation is a subfield inside of natural language understanding that aims to rewrite input text using different wording while maintaining the meaning of sentences. In recent years, several commercial and non-commercial paraphrasers have been developed, including T5, ProtAugment, among others [27, 28, 29]. The Parrot paraphraser is based on the Text-To-Text Transfer Transformer (T5) framework, which was developed by Google in 2019 with the goal of creating a unified framework that converted all text-based language problems into a text-to-text (string) output, as opposed to BERT and its extensions, which typically output a label or a span of the input. Some tasks that T5 may be assigned tasks such as translation, question answering, sentiment analysis, and checking correct grammar, among others [30]. Parrot, in particular, is an adaptation of T5 designed to work on paraphrase generation. Parrot is a framework designed to accelerate natural language understanding models by using data augmentation techniques while maintaining flexibility to control adequacy, fluency, and diversity as needed [31]. The main advantage of this framework is that it preserves the intent of the input text and slots while generating high-quality paraphrases, which other generative models do not guarantee [26].

3) Sample generation based on transitivity property

The final method we will implement to deal with data imbalance is sample generation based on the transitivity property of the data [8]. This data augmentation method is intended for many-to-one paraphrase identification tasks. This method takes advantage of the fact that some questions in the dataset contain multiple other questions that are similar. As a result, if it is known that a question is similar to another and to a third, we can define a transitive relationship between the three as $Q_1 \Leftrightarrow Q_2 \Leftrightarrow Q_3$. In order to exploit this property, we first identified all the questions that had at least two other questions labeled as similar; and then, we defined new combinations between these labeled questions to generate new samples.

C. Approach

To identify which pairs of questions are similar, we use a fine-tuned BERT model as a baseline and estimate the impact of different data augmentation methods on performance to address the class imbalance problem using: (a) focal loss to weight down easy-to-classify pairs of sentences; (b) paraphrase generation using a pre-trained model (Parrot); (c) sample generation based on the transitive property of the dataset. This problem has been extensively researched in the literature [45]. However, to the best of our knowledge, no previous studies have specifically compared class imbalance methods (e.g. using focal loss functions) with data augmentation methods (e.g. paraphraser models and sample generation using properties of the data) on BERT models for paraphrase identification. Furthermore, the goal of this project is to estimate the contribution of various methods to addressing class imbalance and estimate the improvement in performance over the baseline (BERT model). The approach was expected to be successful if access to specialized hardware for computing complex models on potentially large datasets is available. Some researchers compared the computation complexity of various versions of BERT models and concluded that training them requires a large amount of computation of specialized hardware such as Tensor Processing Units (TPUs) and Graphic Processing Units (GPUs) [23]. As a reference, training the base BERT model (~110M parameters) required 12 days of nonstop use of eight V100 GPUs.

One major consideration was balancing performance of the model with training or fine-tuning for QQP by the project deadline. After performing experiments, the raw estimation of the training time for the entire program on a single machine was approximately four (4) days even with access to the Deep Learning Virtual Machines (VM) from Google Cloud Platforms. Given financial constraints, the available hardware was insufficient to properly train the models in time. For this reason, we decided to downsample the dataset by approximately 50%, resulting in 200,000 question pairs to train the baseline. The distribution of the model is 75,310

(63%) of non-duplicated questions and 44,690 (37%) duplicated questions. Nonetheless, the model tuning and data augmentation methods were able to meet the original planning expectations.

We used Pytorch (v. 1.8.1) as the deep learning framework to implement the BERT model. Our implementation was based on the implementation of Verma [44], who used a similar BERT classification model for an entailment task. The focal loss function implemented in our program was developed by another user since it is not directly implemented in Pytorch [48]. The Parrot library (v.1.0) already contained a deployed implementation of the Parrot paraphraser [26]. Finally, the sample generation based on the transitivity property of the data was adapted from Ragav et al. [8].

In terms of the pre-processing of the data, we considered the following scenarios: (a) we removed the punctuation of the questions; (b) we converted all the words to lower; (c) we applied tokenization to create lists of words; (d) we performed stop word removals to maintain only meaningful tokens; (e) we added special tokens to express the beginning of a question and the end of a question; (f) we applied a lemmatization.This pre-processing pipeline for the text was adapted from [8].

The BERT model can accept the tokenized pairs of questions as inputs without limits in the number of characters, and then performs a positional encoding. The model also accepts as input a binary variable that takes the value of one (1) if the question pair is duplicated and zero (0) otherwise. Since this model is a many-to-one classification task, the output of the model will be a binary label indicating if the model predicts the pair of questions are similar or not. We focus on two metrics commonly used in classification tasks to measure success: accuracy and precision. The expression to compute both metrics in a binary classification task is defined in equation 2 and 3 where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives. Note that for this problem the positive label are the duplicated questions. The accuracy is defined as the proportion of samples that are correctly labeled by the model; and the precision is the proportion of the positive predictions that are true positives.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (2)$$

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

We briefly mention the loss functions used in the project before discussing the tuning of the model. The implemented loss function was Cross Entropy, as defined in equation 4, in the case of the base BERT model as well as the models with the augmented samples from paraphrasing and altering the data based on the transitivity property. The loss function for the model that uses focal loss as a strategy to address class imbalance is that shown in equation 1.

$$CE(p_t) = -\log(p_t) \text{ where } p_t = I(y = 1) \quad (4)$$

The first step in optimizing the model was to compare the performance and training speed of the base pre-trained BERT [9] with a pre-trained DistilBERT model [24]. After conducting experiments, we concluded that DistilBERT resulted in approximately 20% faster training at the expense of around 10% decrease in the accuracy metric. As a result, we decided to implement the BERT model given its ability to effectively balance the trade-off between computation complexity and performance. We found graphical and descriptive evidence of model overfitting in previous experiments (e.g.training accuracy increasing from 0.7475 to 0.9821 while validation accuracy leveling off from 0.7915 to 0.8080 in 5 epochs). In other words, given the available sample size, we concluded from previous experiments that training the full BERT model could result in overfitting. To address this issue, we altered the model's architecture by including a dropout layer on the hidden layers, attention heads, and classification heads, as well as weight decay to enforce regularization.

Finally, in order to reduce the model complexity we experimented with adjusting the number of trained parameters. We considered three scenarios: (a) training all of the parameters in the model with the QQP dataset; (b) training only the parameters of the classifier head; (c) training only he last quarter of the model, including the classifier head, given that the first layers of the model generally encode universal features whereas the final layers are more task-specific [46]. The layers in the final quarter of the model include the classifier head, some attention heads and some hidden layers. Ultimately, training only the final quarter of the layers, and incorporating dropout and weight decay, resulted in a significant reduction in overfitting, as well as a 20% reduction in training time due to the smaller number of trained parameters.

Lastly, we performed hyper-parameter tuning. With respect to the batch size, efforts were made to maximize this as much as possible because smaller batch sizes increased training time. However, the maximum batch size was limited by GPU memory due to the large number of parameters in the BERT model. Consequently, we streamed the training data from disk, as opposed to loading all to memory at once, and pre-processed the training data on-the-fly. The batch size was set to 32 to facilitate quick training and was unchanged.The learning rate was varied using the following grid search: {1e-1,1e-2,1e-3,1e-4,1e-5}. However, it did not effect significant improvements on the success metrics. For that reason, we decided to select the learning rate with the best accuracy rate in the validation set, which corresponded to 1e-5. Given the quantity and complexity of the available data, this outcome was anticipated. The computation time was shortened by setting the number of epochs to 5. Finally, we used the AdamW optimizer, an adaptive optimizer widely used for deep learning optimization because it tends to train models to generalize more effectively. AdamW combines improved weight decay regularization with stochastic gradient descent. Lastly, we split the data into 60% training data, 20% validation data, and 20% test data. The complete list of parameters can be found in Table I. The code for this project can be found here. “main model.ipynb” contains the code used for training the model, while “sample transitivity.ipynb” and

"parrot-augmentation.ipynb" contain code for data augmentation.

TABLE I. BASE MODEL PARAMETERS

Parameter	Value
Epochs	5
Learning Rate	1e-5
Pre-trained model	bert-base-uncased
Weight Decay	0.1
Dropout	0.1
Optimizer	AdamW
Trained parameters	Last 25% of BERT layers + classifier head

III. RESULTS AND RESULTS

In this section, we present the results of training the BERT models using various approaches to address the class imbalance problem. We present the success metrics summary results as well as the learning curves for the base model and the different methods in Figure 3 and Table II.

In terms of the training, we observe that the learning curves are consistent with a proper learning of the models, for the loss function as well as the success metrics curves. In other words, after fine-tuning the base model, we do not observe signs of overfitting. This can be seen by comparing the curves for the training and validation accuracies. In all the plots, the training and validation curves have very similar losses and accuracies for all epochs. The training accuracy is not significantly higher than the validation accuracy for all data augmentation methods, which supports the conclusion that the model generalized well for all three data augmentation methods. On the other hand, as mentioned in the approach section, the learning was expected to occur on the last quarter of the network, given that all previous parameters were frozen to avoid overfitting and increase computation times.

Overall, we observe that the fine-tuned baseline model already achieves a good performance of around 87% for validation accuracy and 81% for validation precision. When the methods to address class imbalance are introduced, we observe similar results to the baseline for the model with the focal loss function and the method using the parrot paraphrases. There are several possible explanations for this. The first possible explanation is that the dataset does exhibit some degree of class imbalance (63% of non-duplicated questions and 37% of duplicated questions) but it is not as profound as other tasks in the literature that could exhibit ratios of 90:10 or worse. Given that, it is possible that the model with the focal loss does not dynamically downsample the majority group because it may be correctly predicting some portion of the labels for the minority group, resulting in marginal performance gains. Another possible explanation is that the number of new samples added by the Parrot paraphrases may have been insufficient to boost the performance of the model as exhibited in other experiments [26]. As a reference, using the Parrot paraphraser we added around 4% more samples. The reason to maintain a low number of new samples was to avoid the risk of underfitting the real non-generated data, as well as the large computation times of the paraphrases which generated around 100 samples every 30 minutes on specialized hardware.

The sample generation based on the transitivity property of the data provided an improvement to model performance of around 4% of the validation precision. This data augmentation method led to an increase of around 32% in the sample size mostly from duplicated questions. This led to a balanced dataset with a ratio of 50:50 (125,524 duplicated questions or 51%, and 118,646 or 49% non-duplicated questions). Because this data augmentation method generates samples consisting of real questions, but in different pairs, it is likely that it generated sufficient meaningful new samples to balance the dataset and improve the ability of the model to predict duplicated questions. In future research, we recommend analyzing the impact of different ratios in the augmented dataset using the different methods from this project, in order to analyze the performance change.

TABLE II. TRAIN AND VALIDATION RESULTS

Model	Train accuracy	Validation accuracy	Train precision	Validation precision
Baseline	0.8934	0.8696	0.8397	0.8123
Baseline + (1)	0.8714	0.8636	0.8125	0.8001
Baseline + (2)	0.8913	0.8666	0.8379	0.8112
Baseline + (3)	0.9086	0.8838	0.8887	0.8449

NOTE: (1) REPRESENTS THE BERT MODEL WITH FOCAL LOSS FUNCTION; (2) REPRESENTS THE BERT MODEL WITH THE SAMPLES AUGMENTED USING THE PARROT PARAPHRASER; (3) REPRESENTS THE BERT MODEL WITH THE SAMPLES GENERATED USING THE TRANSITIVITY PROPERTY OF THE DATA.

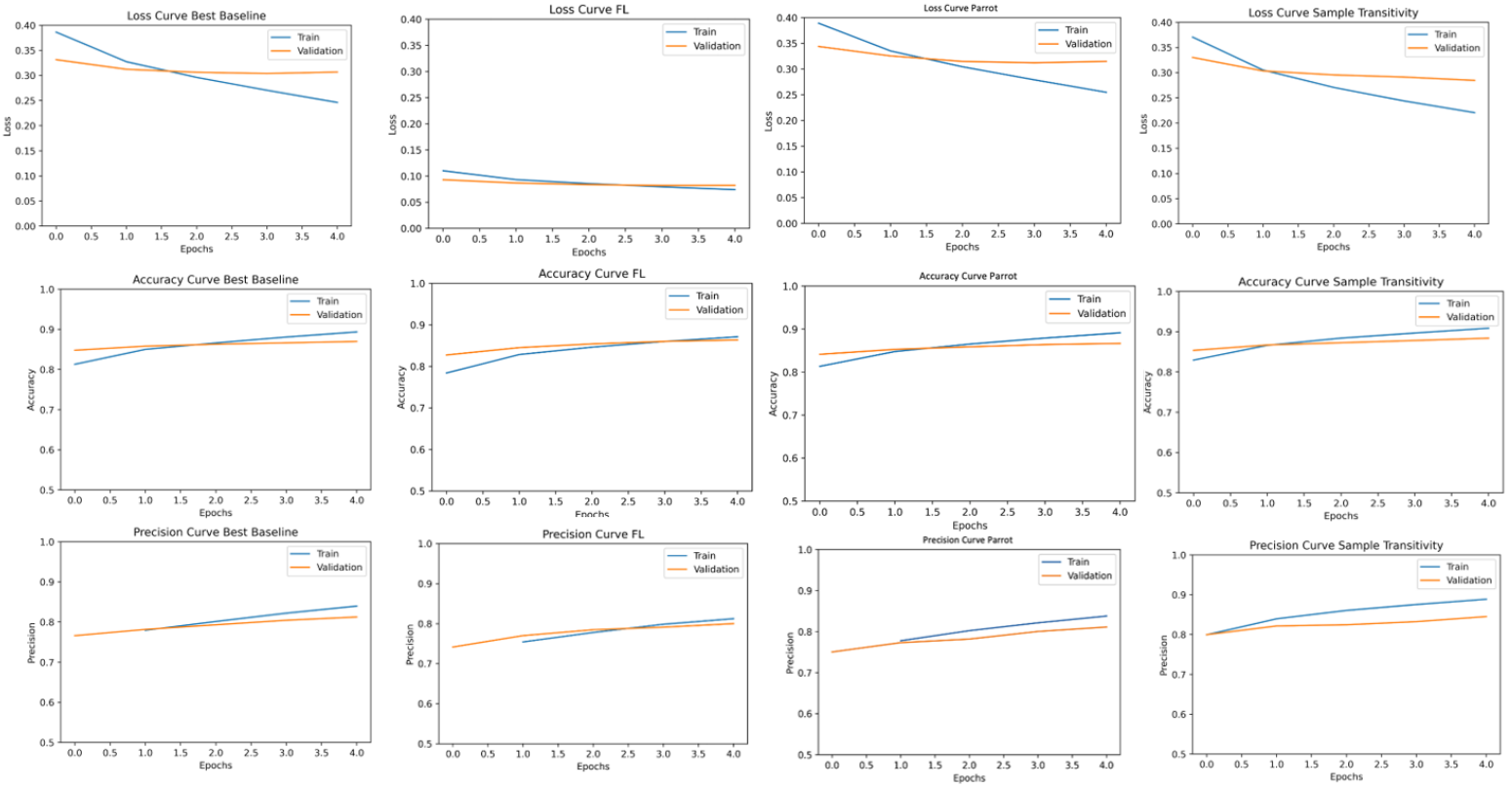

FIGURE 3. TRAINING CURVES. OWN CALCULATION

IV. WORK DIVISION

A summary of the contributions is provided in Table X. Additionally, the team held bi-weekly meetings to discuss aspects of the projects. All members reviewed the paper write-up and the code from the other team members.

TABLE III. WORK DIVISION

Student	Contributed aspects	Details
Anubhaw Mathur	Data augmentation methods, model fine-tuning report writing	Wrote the code to implement one data augmentation method, fine-tuned the model, and wrote section III of the report
Simon Tong	Model implementation and fine-tuning, report review, infrastructure setup	Set up infrastructure for model training and wrote the code to implement the BERT classification problem
Alejandro Vasquez Quijada	Implementation of data augmentation models and report writing	Wrote the code to implement two data augmentation methods, fine-tuned the model and wrote sections I and II in the report

V. REFERENCES

[1]T. Schleifer, "Question-and-answer site Quora still exists, and it's now worth $2B," Vox, May 16, 2019. https://www.vox.com/recode/2019/5/16/18627157/quora-value-billion-question-answer

[2]Kaggle, "Quora Question Pairs," kaggle.com, 2017. https://www.kaggle.com/c/quora-question-pairs

[3]Y. Zhang, J. Baldridge, and L. He, "PAWS: Paraphrase Adversaries from Word Scrambling," arXiv:1904.01130 [cs], Apr. 2019, Accessed: Jul. 30, 2022. [Online]. Available: https://arxiv.org/abs/1904.01130

[4]Y. He, Z. Wang, Y. Zhang, R. Huang, and J. Caverlee, "PARADE: A New Dataset for Paraphrase Identification Requiring Computer Science Domain Knowledge," arXiv:2010.03725 [cs], Oct. 2020, Accessed: Jul. 30, 2022. [Online]. Available: https://arxiv.org/abs/2010.03725

[5]W. B. Dolan and C. Brockett, "Automatically Constructing a Corpus of Sentential Paraphrases," ACLWeb, 2005. https://aclanthology.org/I05-5002/ (accessed Jul. 30, 2022).

[6]R. Longadge and S. Dongre, "Class Imbalance Problem in Data Mining Review," arXiv:1305.1707 [cs], May 2013, Accessed: Jul. 30, 2022. [Online]. Available: https://arxiv.org/abs/1305.1707

[7]P. Damodaran, "Parrot: Paraphrase Generation for NLU.," pythonlang.dev, 2021. https://pythonlang.dev/repo/prithivirajdamodaran-parrot_paraphraser/ (accessed Jun. 22, 2022).

[8]A. Ragav, A. Sekar, N. Narayanan, P. Guruprasad, and Y. Jakhotiya, "Can we avoid repeated questions by identifying similar intent?," Blog, Oct. 03, 2021.

https://yashjakhotiya.github.io/blog/nlp/2021/10/03/quora-question-pairs.html (accessed Jun. 22, 2022).

[9]J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv.org, Oct. 11, 2018. https://arxiv.org/abs/1810.04805

[10]J. Briggs, "BERT For Measuring Text Similarity," Medium, Sep. 02, 2021. https://towardsdatascience.com/bert-for-measuring-text-similarity-eec91c6bf9e1 (accessed Jul. 30, 2022).

[11]Y. Arase and J. Tsujii, "Transfer fine-tuning of BERT with phrasal paraphrases," Computer Speech & Language, vol. 66, p. 101164, Mar. 2021, doi: 10.1016/j.csl.2020.101164.

[12]B. Ko and H.-J. Choi, "Twice fine-tuning deep neural networks for paraphrase identification," Electronics Letters, vol. 56, no. 9, pp. 444–447, Apr. 2020, doi: 10.1049/el.2019.4183.

[13]S. Iyer, N. Dandekar, and K. Csernai, "First Quora Dataset Release: Question Pairs," Quora, 2021. https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs

[14]Papers with Code, "Quora Question Pairs Dataset," paperswithcode.com, 2022. https://paperswithcode.com/dataset/quora-question-pairs (accessed Jun. 19, 2022).

[15]A. P. Parikh, O. Täckström, D. Das, and J. Uszkoreit, "A Decomposable Attention Model for Natural Language Inference," arXiv:1606.01933 [cs], Sep. 2016, [Online]. Available: https://arxiv.org/abs/1606.01933

[16]Q. Chen, X. Zhu, Z. Ling, S. Wei, H. Jiang, and D. Inkpen, "Enhanced LSTM for Natural Language Inference," Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1657–1668, 2017, doi: 10.18653/v1/P17-1152.

[17]E. Cohen, "How to predict Quora Question Pairs using Siamese Manhattan LSTM," Medium, Sep. 16, 2018. https://blog.mlreview.com/implementing-malstm-on-kaggles-quora-question-pairs-competition-8b31b0b16a07 (accessed Jul. 30, 2022).

[18]M. HONNIBAL, "Deep text-pair classification with Quora's 2017 question dataset · Explosion," Explosion.AI, Feb. 12, 2017. https://explosion.ai/blog/quora-deep-text-pair-classification (accessed Jun. 19, 2022).

[19]Z. Wang, W. Hamza, and R. Florian, "Bilateral Multi-Perspective Matching for Natural Language Sentences," arXiv:1702.03814 [cs], Jul. 2017, [Online]. Available: https://arxiv.org/abs/1702.03814

[20]P. He, X. Liu, J. Gao, and W. Chen, "DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION," openreview.net, Sep. 28, 2020. https://openreview.net/forum?id=XPZIaotutsD (accessed Jun. 20, 2022).

[21]Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations," openreview.net, Sep. 25, 2019. https://openreview.net/forum?id=H1eA7AEtvS (accessed Jun. 20, 2022).

[22]Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized Autoregressive Pretraining for Language Understanding," arXiv.org, 2019. https://arxiv.org/abs/1906.08237

[23]Suleiman Khan, Ph.D, "BERT, RoBERTa, DistilBERT, XLNet—which one to use?," Medium, Sep. 04, 2019. https://towardsdatascience.com/bert-roberta-distilbert-xlnet-which-one-to-use-3d5ab82ba5f8

[24]V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," arXiv.org, 2019. https://arxiv.org/abs/1910.01108

[25]B. Muller, "BERT 101 - State Of The Art NLP Model Explained," huggingface.co, Mar. 2022. https://huggingface.co/blog/bert-101

[26]Prithivida, "Parrot Paraphraser," GitHub, Jul. 30, 2022. https://github.com/PrithivirajDamodaran/Parrot_Paraphraser (accessed Jul. 31, 2022).

[27]Jeff, "Creating a Paraphrase Generator Model using T5 and Deploying on Ainize," Medium, Dec. 02, 2021. https://medium.com/@imjeffhi4/creating-a-paraphrase-generator-model-using-t5-and-deploying-on-ainize-7742bc83532a (accessed Jul. 31, 2022).

[28]K.-H. Huang and K.-W. Chang, "Generating Syntactically Controlled Paraphrases without Using Annotated Parallel Pairs," arXiv:2101.10579 [cs], Jan. 2021, Accessed: Jul. 31, 2022. [Online]. Available: https://arxiv.org/abs/2101.10579

[29]K. Ding et al., "Learning to Selectively Learn for Weakly-supervised Paraphrase Generation," arXiv:2109.12457 [cs], Sep. 2021, Accessed: Jul. 31, 2022. [Online]. Available: https://arxiv.org/abs/2109.12457

[30]H. Digaari, "Solved! Google's Text-To-Text Transfer Transformer (T5) Bottleneck," Medium, Mar. 08, 2021. https://towardsdatascience.com/hands-on-googles-text-to-text-transfer-transformer-t5-with-spark-nlp-6f7db75cecff

[31]C. Liu, D. Dahlmeier, and H. T. Ng, "PEM: A paraphrase evaluation metric exploiting parallel texts," scholarbank.nus.edu.sg, 2010. https://scholarbank.nus.edu.sg/handle/10635/40611 (accessed Aug. 01, 2022).

[32]J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," Journal of Big Data, vol. 6, no. 1, Mar. 2019, doi: 10.1186/s40537-019-0192-5.

[33]L. Yu and N. Zhou, "Survey of Imbalanced Data Methodologies," arXiv:2104.02240 [cs, stat], Apr. 2021, Accessed: Aug. 01, 2022. [Online]. Available: https://arxiv.org/abs/2104.02240

[34]B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," Progress in Artificial Intelligence, vol. 5, no. 4, pp. 221–232, Apr. 2016, doi: 10.1007/s13748-016-0094-0.

[35]H. Lee, M. Park, and J. Kim, "Plankton classification on imbalanced large scale database via convolutional neural networks with transfer learning," IEEE Xplore, Sep. 01, 2016. https://ieeexplore.ieee.org/document/7533053 (accessed Aug. 01, 2022).

[36]S. Pouyanfar et al., "Dynamic Sampling in Convolutional Neural Networks for Imbalanced Data Classification," IEEE Xplore, Apr. 01, 2018. https://ieeexplore.ieee.org/document/8396983/ (accessed Jun. 24, 2020).

[37]M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," Neural Networks, vol. 106, pp. 249–259, Oct. 2018, doi: 10.1016/j.neunet.2018.07.011.

[38]S. Wang, W. Liu, J. Wu, L. Cao, Q. Meng, and P. J. Kennedy, "Training deep neural networks on imbalanced data sets," 2016 International Joint Conference on Neural Networks (IJCNN), Jul. 2016, doi: 10.1109/ijcnn.2016.7727770.

[39]T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 1–1, 2018, doi: 10.1109/tpami.2018.2858826.

[40]H. Wang, Z. Cui, Y. Chen, M. Avidan, A. B. Abdallah, and A. Kronzer, "Predicting Hospital Readmission via Cost-Sensitive Deep Learning," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 15, no. 6, pp. 1968–1978, Nov. 2018, doi: 10.1109/tcbb.2018.2827029.

[41]S. H. Khan, M. Hayat, M. Bennamoun, F. Sohel, and R. Togneri, "Cost Sensitive Learning of Deep Feature Representations from Imbalanced Data," arXiv:1508.03422 [cs], Mar. 2017, Accessed: Aug. 04, 2022. [Online]. Available: https://arxiv.org/abs/1508.03422

[42]S. Ando and C. Y. Huang, "Deep Over-sampling Framework for Classifying Imbalanced Data," Machine Learning and Knowledge Discovery in Databases, pp. 770–785, 2017, doi: 10.1007/978-3-319-71249-9_46.

[43]C. Huang, Y. Li, C. C. Loy, and X. Tang, "Learning Deep Representation for Imbalanced Classification," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016, doi: 10.1109/cvpr.2016.580.

[44]D. Verma, “Fine-tuning pre-trained transformer models for sentence entailment,” Medium, Jan. 15, 2021. https://towardsdatascience.com/fine-tuning-pre-trained-transformer-models-for-sentence-entailment-d87caf9ec9db (accessed Aug. 04, 2022).

[45]M. Bayer, M.-A. Kaufhold, and C. Reuter, “A Survey on Data Augmentation for Text Classification,” ACM Computing Surveys, p. 3544558, Jun. 2022, doi: 10.1145/3544558.

[46] Raphaël B. “How many layers of my BERT model should I freeze?” raphaelb.org, May 10, 2021. https://raphaelb.org/posts/freezing-bert/