Data Augmentation with Feedback Control for BERT Multitask Finetuning

Stanford CS224N Default Project

Kevin Supakkul and Ryan Beauchamp
Department of Computer Science
Stanford University
supakkul@stanford.edu, rmb87@stanford.edu

Abstract

Data augmentation (DA) is a popular and effective technique in computer vision. Can it be effectively used as part of multitask finetuning to help large language models (LLMs) such as BERT better generalize to unseen data? While DA approaches exist for natural language processing (NLP), their benefit for multitask finetuning of LLMs is underexplored. We introduced data augmentation with feedback control (DAFC) to generate robust data augmentation for multitask finetuning LLMs, with the goal of improving performance for three common NLP tasks that include sentiment analysis (SA), paraphrase detection (PD), and semantic textual similarity (STS). Benefits of DAFC include 1) a feedback loop that verifies augmented sentences are sufficiently similar to the original sentences, 2) multiple rounds of augmentation that result in more diverse sentences, 3) the ability to generate a variety of training data leads to a variety of models, which results in an ensemble that empirically improves performance. Our experiments demonstrated that DAFC improved the performance of multitask finetuning, even over existing DA approaches. By using DAFC, the multitask model outperformed a model that did not use DA by 1.8%. The performance gain from DAFC increased further with ensembles, for which the DAFC ensemble scored 2.5% higher overall than the non-DA ensemble. Finally, DAFC also outperformed vanilla DA, with BT improving 1.0% and EDA improving 2.4% with the DAFC method.

1 Introduction

Data Augmentation (DA) is a popular approach for improving the ability of neural networks to generalize to unseen data. However, most of the prior literature on DA focused on computer vision. While applying DA to natural language processing (NLP) has been explored, it is a relatively new field with low usage and limited success (Pellicer et al., 2023). One successful method that has been applied is easy data augmentation (EDA), but its authors highlighted that it is unlikely that large language models (LLMs) would benefit from EDA (Wei and Zou, 2019). Since LLMs, such as BERT, are at the forefront of NLP today, finding a DA method that helps LLMs better generalize to unseen data was our focus with this paper. We theorized that one problem with existing approaches to DA for NLP is that the techniques are open-loop, meaning that the data generation contains a forward pass but no error-checking to verify that the new sentence is a useful one. This is especially concerning for NLP since sentence semantics are highly sensitive to noise (Al Sharou et al., 2021). To solve this open-loop concern, we adapted ideas from the field of feedback control (FC), a highly established field of study in hardware-related disciplines such as robotics (Franklin et al., 2014). FC involves measuring the real-world error of the control algorithm output and using that information to modify the next input to the control algorithm. In this paper, we introduce data augmentation with feedback control (which we'll refer to as DAFC) with the goal of improving performance for three common NLP tasks, including sentiment analysis (SA), paraphrase detection (PD), and semantic textual similarity (STS).

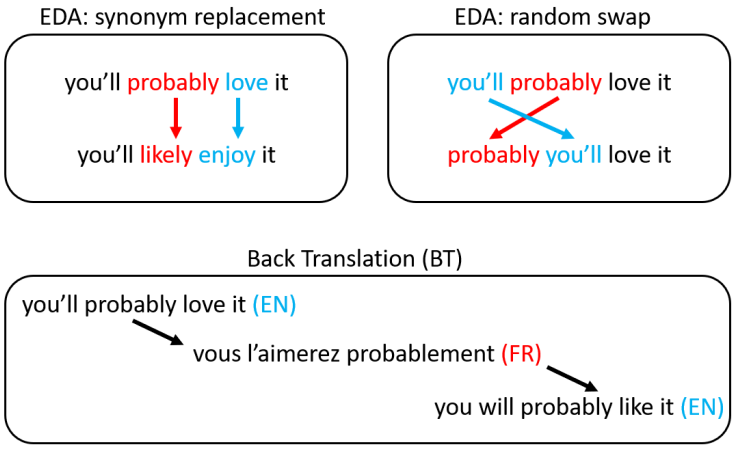

Figure 1: Existing data augmentation (DA) techniques include easy data augmentation (EDA) and backtranslation (BT).

2 Related Work

The most common methods of DA for NLP were explored in depth in Li et al. (2022), including methods that we incorporated in our approach such as paraphrasing (e.g. synonym replacement, backtranslation) and noising techniques (e.g. random swap, random deletion). The paper, however, does not include a quantitative analysis of which techniques perform best, so it was left to our own experimentation to discover the best path forward.

As mentioned in Section 1, EDA is a technique that combines paraphrasing and noising approaches (Wei and Zou, 2019). EDA involves randomly applying one of four operations on the original sentence: 1) synonym replacement (SR) - randomly replace n non-stop words with synonyms, 2) random insertion (RI) - randomly insert synonyms for n non-stop words, 3) random swap (RS) - randomly select two words and swap them n times, 4) random deletion (RD) - randomly remove each word in the sentence with probability p. For the hyperparameters, we used n = len(sentence) * alpha and p = alpha, with alpha = 0.2 for SR and RD and 0.1 for RI and RS, as these were the highest performing hyperparameters in the original paper. However, the authors suggest that LLMs such as BERT are unlikely to benefit from EDA. In this paper, we test this claim and also propose enhancing EDA with DAFC for use with LLMs.

Feedback control (FC), our underlying enhancement to DA, is well-documented and widely used in hardware-adjacent disciplines such as robotics. The main idea of feedback control is to measure the real-world output of an algorithm and feed the measurement back into the algorithm to help it achieve some desired output (Franklin et al., 2014). Figure 2 shows an example of a feedback control block diagram. We used this idea of FC to improve the dynamics of DA for NLP as part of our DAFC approach in 3.1.

Before testing our DAFC approach, we first aimed to establish a strong baseline performance using insights from existing literature. Some beneficial existing approaches for multitask finetuning include cosine-similarity, contrastive learning, and summing losses across multiple tasks.

Cosine-similarity is a similarity metric that returns a continuous output that is useful for both regression and classification tasks. It treats embeddings as vectors in multidimensional space and computes a cosine of the angle between them, returning a normalized output between 0 (least similar) and 1 (most similar) (Reimers and Gurevych, 2019).

$$\text{SimCSE training objective} = \ell_i = -\log \frac{e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N} \left(e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j^+)/\tau} + e^{\text{sim}(\mathbf{h}_i, \mathbf{h}_j^-)/\tau} \right)} \quad (1)$$

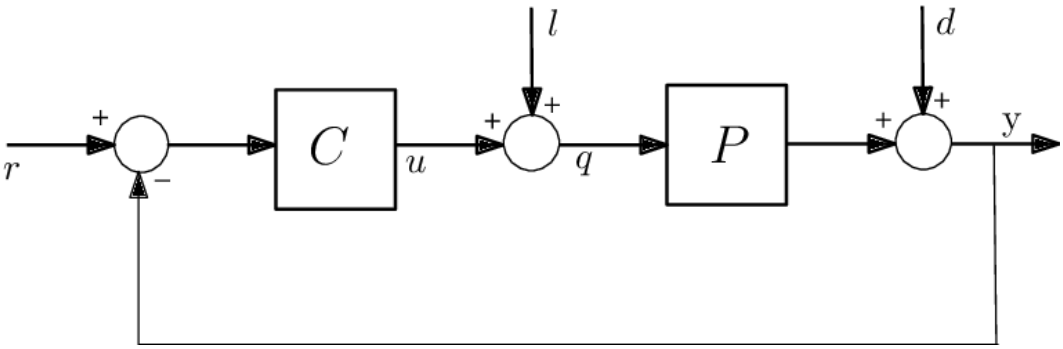

Figure 2: Block diagram of classical feedback control, where r is the desired value, C is the control algorithm, P is the real-world plant dynamics, and y is the measured output (Pandey et al., 2018)

Gao et al. (2021) describe simple contrastive learning approaches for pretraining BERT to improve performance on STS, which they claimed can achieve a Spearman's correlation of 86.2 on the STS-B development set using $BERT_{base}$. Following the approach that drove their best performance, we pretrained on positive (entailment) and negative (contradiction) sentence pairs from the the SNLI and MNLI datasets, with the goal to have the BERT embeddings have a higher cosine similarity in the positive case and lower cosine similarity in the negative case. The chosen loss function ℓ_i for each example i is shown in equation 1 where sim is the cosine similarity function and τ is the temperature hyperparameter, for which we used 0.05 based on the paper.

$$\text{Loss}_{train} = \text{Loss}_{SA} + \text{Loss}_{PD} + \text{Loss}_{STS} \tag{2}$$

In order to incorporate gradients for all tasks, we utilized the multitask finetuning method described in Bi et al. (2022). Specifically, we fine-tuned BERT using a summation of losses from all three tasks, as shown in equation 2.

3 Approach

First, we pretrained the model using simple contrastive learning from Gao et al. (2021) as described in Section 2.

Next, we finetuned the model on the SA, PD, and STS tasks. For SA, we used a dropout layer and a linear layer that maps BERT embeddings to 5 classes, and used cross entropy loss during training. For PD and STS, we used BERT to compute embeddings for each sentence pair, then used cosine similarity to produce a similarity score. For PD, this score was used directly for predicting its binary labels, and for STS, we scaled this score to be in the range of 0-5 to predict its labels. During training, PD and STS loss was computed with MSELoss. We summed all three losses as described in equation 2, then backpropagated the gradients of this combined loss function.

In order to compare our approach with DAFC and without DA, we used the same model architecture with the same hyperparameters. We tested the addition of DAFC to the finetuning step, using DAFC to generate more data for the SA, PD, and STS tasks. Our specific DAFC algorithm is described in more detail below.

3.1 Data Augmentation with Feedback Control (DAFC)

Our DA approach leverages existing literature: EDA by Wei and Zou (2019) and BT using the M2M100 NMT model from Fan et al. (2020). Our contributions with DAFC are 1) the addition of a feedback control loop to confirm the augmented sentence is sufficiently similar to the original sentence, 2) using the augmented sentence to then produce subsequent, further augmented sentences, and 3) randomly generating multiple finetuned models with DAFC to create an ensemble that better generalizes to unseen data.

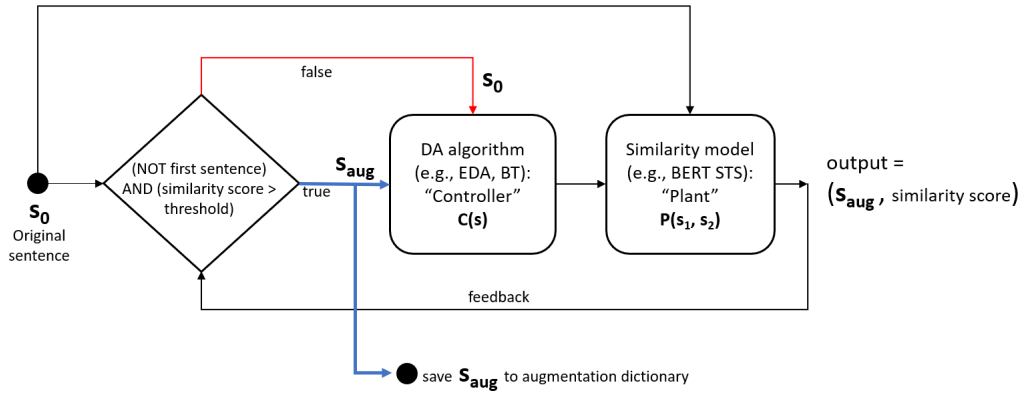

Figure 3: Block diagram of data augmentation with feedback control (DAFC) for NLP. This algorithm can be executed for as many iterations (loops) as necessary to generate a robust dataset of augmentations.

Figure 3 illustrates the flow of data in this feedback loop. The process starts with some original sentence, which is passed through an augmentation algorithm. For the augmentation algorithm, one can use any existing DA algorithm such as EDA, BT, or even augmented sentences generated by ChatGPT. After generating a new sentence, that sentence is passed to the similarity model which computes a similarity score comparing the new sentence to the original. In our experimentation, we chose the similarity model to be our BERT model for the STS task. If the similarity score is above a defined threshold, then the augmented sentence is predicted to be sufficiently similar to the original sentence so this new sentence is saved in a dictionary of possible augmentations for multitask finetuning. Then this valid new sentence is passed back into the pipeline, feeding it into the augmentation algorithm to generate a new augmented sentence. However, if the similarity score is insufficient, then the new sentence is discarded so the pipeline uses the original sentence as the next input into the augmentation algorithm. This feedback loop can be executed for as many iterations as desired, and the quantity of iterations is a hyperparameter that can be tuned. Section 5 includes an ablation study involving this hyperparameter.

There are several important ideas to keep in mind for this approach. First, the augmentation algorithm must have some randomness in order to generate novel sentences in the case that a previous augmentation was insufficient. Second, it is important to use a high quality similarity model to ensure that the similarity scores are reliable. Third, this process is better performed offline due to the latency of computation, especially if the similarity model is a deep neural network.

4 Experiments

We ran a controlled experiment to compare multitask finetuning with DAFC and without DA. We also compared our DAFC approach to the vanilla EDA and BT methods.

4.1 Data

For SA, we used the Stanford Sentiment Treebank dataset consisting of 11,855 single sentences from movie reviews and a label indicating negative, somewhat negative, neutral, somewhat positive, or positive (Socher et al., 2013). For PD, we used the Quora dataset consisting of 202,152 pairs of questions and a label indicating whether they are paraphrases of one another (SambitSekhar, 2017). For STS, we used the SemEval STS Benchmark dataset consisting of 8,628 sentence pairs with labels indicating their similarity from 0 (unrelated) to 5 (equivalent meaning) (Agirre et al., 2012).

In addition to the data used for training and evaluation, we used additional datasets for SimCSE pretraining. Specifically, we followed the recommendation of Gao et al. (2021) and used the SNLI and MNLI datasets, which together contain 314,000 pairs of entailments and contradictions.

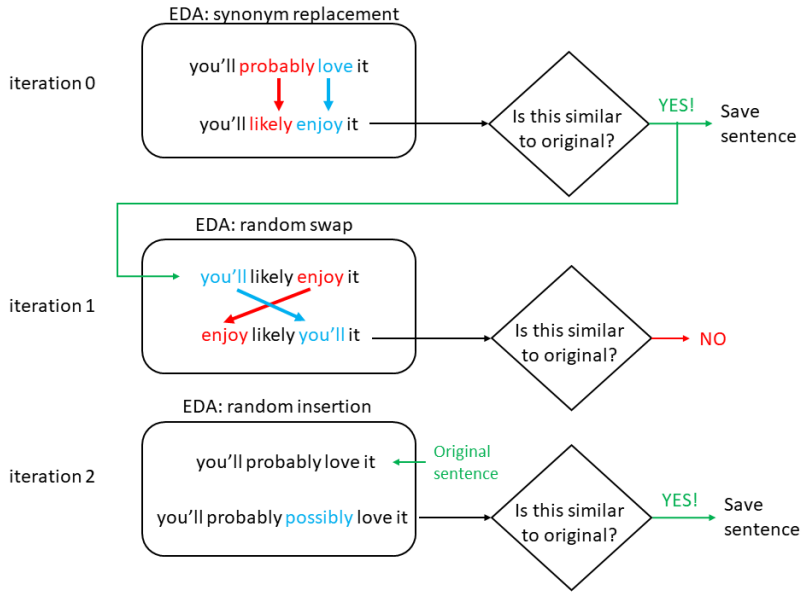

Figure 4: Example of how DAFC manipulates sentences using EDA as the DA algorithm "controller." 5 iterations performed best in our hyperparameter search.

4.2 Evaluation method

For SA and PD, we computed the accuracy of our predictions relative to the true labels. For STS, we computed the Pearson correlation of the predicted similarity values against the true similarity values. Since we are focused on building a multitask model that can predict all three tasks with the same embeddings, we also care about the mean across all three metrics.

4.3 Experimental details

In order to control the experiment, we kept hyperparameters constant for testing with DAFC and without DA. We used a learning rate of 0.00001, AdamW optimizer, weight decay of 0.01, dropout of 0.1, and a step learning rate scheduler with step of 5 and $\gamma = 0.1$. Due to the computation requirements of the massive datasets and a tight project timeline, we had to limit the number of epochs to 7 for each model. Each batch contained data for all three tasks, so each epoch passed through all the data for all tasks (passing through the SA and STS datasets multiple times since they are smaller). The memory constraints of our RTX 3090 GPU limited us to a batch size of 32 during training, and each epoch took 25 minutes to train. After all models finished training, we generated ensembles of three feedback-augmented models as well as three non-augmented models, using aggregate voting for PD and average scores for SA and STS.

After running controlled experiments to measure the effect of DAFC, we also wanted to measure how well an ensemble of data-augmented generalist models (i.e. models that can perform all three tasks) could compare to an ensemble of specialist models without DA (i.e. models that excel at a single task), since an ensemble of specialists often outperform an ensemble of generalists (Meyen et al., 2021). The aim here is to show that a generalist multitask models supplemented by DAFC can perform respectably in comparison to an ensemble of single-task specialists without DA. For this comparison, we performed thorough hyperparameter tuning (HT) for both the specialist and generalist models, which involved setting up an automated script to perform grid search over the hyperparameters above and training for up to 15 epochs. As part of this, we determined that 5 iterations of DAFC performed best relative to 1 or 15 iterations (see Figure 4 for a visual of DAFC iterations).

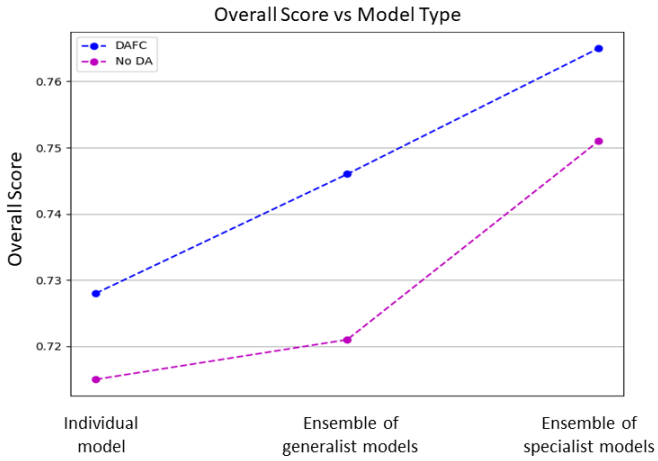

Figure 5: Plot comparing models trained with DAFC and with no DA. One can see that models trained with DAFC perform noticeably better overall

4.4 Results

Model	Development Set					Test Set	
	Score	SA	PD	STS	Rank	Score	Rank
Pretrained BERT Embeddings	0.38	0.38	0.64	0.12	#150 (16%)	NA	NA
Baseline: Multitask w/o DA	0.715	0.494	0.837	0.815	#47 (74%)	NA	NA
Multitask + vanilla EDA	0.707	0.479	0.813	0.830	#50 (72%)	NA	NA
Multitask + vanilla BT	0.721	0.504	0.821	0.839	#41 (77%)	NA	NA
Multitask + DAFC (EDA)	0.724	0.508	0.839	0.818	#41 (77%)	NA	NA
Multitask + DAFC (BT)	0.728	0.526	0.829	0.830	#38 (79%)	NA	NA
Generalist ensemble w/o DA	0.721	0.502	0.844	0.818	#41 (77%)	NA	NA
Generalist ensemble + DAFC (EDA/BT)	0.739	0.533	0.849	0.834	#31 (83%)	NA	NA
Generalist ensemble + HT + DAFC (EDA/BT)	0.746	0.532	0.859	0.846	#24 (87%)	0.734	#25 (80%)
Specialist ensemble + HT w/o DA	0.751	0.526	0.861	0.867	#21 (88%)	NA	NA
Specialist ensemble + HT + DAFC (EDA/BT)	0.765	0.547	0.877	0.872	#4 (98%)	0.760	#10 (92%)

Table 1: Model performance on the SA, PD, and STS development and test sets, including relative leaderboard rankings as of 12:00AM March 19, 2023.

Table 1 shows the results of experimentation. One can see that by using DAFC, the multitask model outperforms a model that does not use DAFC by .013 (1.8%). This is especially true when considering ensembles, for which DA models score 0.018 (2.5%) higher overall than non-DA models, as illustrated in Figure 5. Something curious that we did not expect is that adding vanilla EDA as described in Wei and Zou (2019) hurt performance. On the contrary, vanilla BT improves performance over the non-DA model and nearly matches the performance of DAFC with BT.

Of the DA techniques, EDA benefits more from DAFC than BT, with EDA improving 0.017 (2.4%) compared to BT's improvement of 0.007 (1.0%). Another interesting result is that an ensemble of generalists using DAFC approaches the performance of an ensemble of specialists that do not use DA. More in-depth experimentation is needed to verify statistical significance of the improvements DAFC is driving, but early results are promising.

5 Analysis

Our experiment demonstrated the ability of DAFC to improve the performance of multitask finetuning. In general, DA can provide a positive benefit on neural networks by introducing more diversity into the dataset, allowing better generalization to unseen data (Wei and Zou, 2019). For BERT multitask

finetuning, why is DAFC more effective than standalone DA approaches such as EDA and BT? Standalone DA algorithms may produce incoherent sentences with poor diction; this is especially true for EDA which can, for example, substitute words with completely irrelevant meaning in the context of the sentence. Additionally, noise from poor augmentation is further compounded in multitask finetuning; an erroneous gradient for one task affects not just itself, but also impacts all other tasks (this helps explain why EDA performs worse than the baseline in multitask finetuning, even though EDA has been shown in prior literature to improve individual tasks). By providing error-checking, DAFC is able to filter out irrelevant sentences. Furthermore, this then makes it possible for DAFC to go through multiple rounds of augmentation, ultimately producing more relevant diverse input for the multitask training. Finally, the randomization of DAFC provides a suitable method for producing multiple models to make up an ensemble, which saw the highest performance boost from DAFC in our experiments.

In our experiments, SA was the most challenging task, scoring significantly lower across the board than the PD and STS tasks. Part of this is due to accuracy not capturing near misses in the ratings scale. In many cases, the model's predicted scores were off by 1, yet closeness isn't taken into account with accuracy. For some development set examples, the model made more significant errors. For example, for the movie review "A deep and meaningful film" the true label was 4 (positive) but the Generalist ensemble + HT + DAFC (EDA/BT) predicted 1 (somewhat negative). One explanation is there are only a few words to work off of in this short review, and the words are somewhat ambiguous. For example, "deep" appears in 81 training examples that range from 1 (somewhat negative) to 4 (positive), so the model may have had difficulty differentiating these cases. Furthermore, "meaningful" appeared in only 6 training examples, all of which were below 4, the true label in this case. DAFC can help here by providing a richer understanding of the words "deep" and "meaningful" via synonym replacement, but it can only do so much with such a short review.

5.1 Ablation study

		DAFC score		
DA algorithm	vanilla score	1 iteration	5 iterations	15 iterations
EDA	0.707	0.715	0.724	0.711
BT	0.721	0.727	0.728	N/A

Table 2: Ablation study testing the impact of the number of DAFC iterations on the score. We see that just 1 iteration is an improvement over the vanilla implementation, and adding more iterations further improves performance. However, too many iterations can hurt performance too, as seen for EDA.

In analyzing our DAFC approach, we performed an ablation study to quantify the benefit of multiple iterations of feedback. With the same hyperparameters that we trained on the vanilla implementations of EDA and BT, we trained models where we modified how many iterations of feedback were used during DAFC. To elaborate, 1 iteration in DAFC means that the original sentence is augmented once, and the augmented sentence is saved if and only if it is sufficiently similar to the original. More than 1 iteration means that a valid augmented sentence is fed back as an input into the DAFC algorithm to generate more sentences. As seen in Table 2, using just 1 iteration improved both EDA and BT. When adding 5 iterations of feedback, both EDA and BT improved, with EDA showing a much larger improvement. This makes sense intuitively, since each iteration of EDA has the potential to change the original sentence drastically; therefore, an augmented sentence that has 5 iterations of EDA has the potential to become more differentiated relative to just 1 iteration of EDA. However, an interesting finding is that 15 iterations of feedback severely hurt the performance of EDA, suggesting that at some point the augmentations compound too much (note that we did not perform 15 iterations of BT due to the computation time that requires). Overall, the feedback loop is beneficial, and when set to an optimal number of iterations DAFC can notably outperform existing DA approaches.

6 Conclusion

We introduced data augmentation with feedback control (DAFC) to generate robust data augmentation for finetuning LLMs, and showed that BERT multitask finetuned with DAFC resulted in improved performance for the SA, PD, and STS tasks. In particular, by using DAFC, the multitask model

outperformed a model that does not use DA by 0.013 (1.8%), and for ensembles, DAFC models scored 0.018 (2.5%) higher overall. Finally, DAFC also outperformed vanilla DA, with BT improving 0.007 (1.0%) and EDA improving 0.017 (2.4%) with the DAFC method. Future work includes confirming this data holds with statistical significance, and extending this approach to other LLMs beyond BERT.

6.1 Limitations

We cannot conclude with statistical significance the performance gains achieved from DAFC. In order to do so, we would need to randomly generate many ensembles using DAFC and measure their results in aggregate to confirm the performance improvements hold with statistical significance. In addition, it would be worthwhile to re-implement the DAFC approach for other LLMs to to confirm the performance gains from DAFC extend beyond BERT.

Another limitation is the computation cost of DAFC. Since we use a transformer to check for viable sentences, generating augmentations takes much longer with DAFC than with a quick online approach such as EDA. For example, an RTX 3090 GPU takes about 30 hours to generate 5 iterations of DAFC with BT for the three training datasets we used. At the same time, the same task for EDA took only 3 hours, so the time requirements are sensitive to the data augmentation algorithm used.

6.2 Future Work

While this project is a good starting point and shows that DAFC is a promising approach, there is a more work we can do to further improve this framework. First, we can improve the feedback loop by using a better similarity "plant" model, as illustrated in Figure 3. The current results are based on a similarity model that we trained ourselves; we simply used our own model finetuned on the STS task, and also experimented with using our own model finetuned on the PD task as the plant model. There exist more complex state of the art models that can perform paraphrase detection such as ChatGPT, so we plan to explore substituting our existing plant model with those high performing models. Another idea for further experimentation is to provide more fine-grained control of augmentation parameters via the feedback loop. Right now we have a binary response to feedback (choose which sentence to feed into the augmentation algorithm), but perhaps a more effective implementation would be to update hyperparameters of the augmentation algorithm (such as α for EDA) using a continuous similarity score. Additionally, we need to train a larger sample size of models so that we can calculate whether DAFC is a statistically significant improvement over existing approaches.

Note that for the coding development of this work, we implemented by-hand each of the methods from our model that are described in this paper (including but not limited to DAFC, SimCSE, multitask finetuning, ensembling), with the exception of the M2M100 translation transformer for which we used the implementation from Fan et al. (2020) and some code in the BERT model which was provided by the Stanford course CS224N. Our novel contribution to NLP is the DAFC algorithm, which as far as we know is original.

References

Eneko Agirre, Johan Bos, Mona Diab, Suresh Manandhar, Yuval Marton, and Deniz Yuret. 2012. * sem 2012: The first joint conference on lexical and computational semantics–volume 1: Proceedings of the main conference and the shared task, and volume 2: Proceedings of the sixth international workshop on semantic evaluation (semeval 2012). In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*.

Khetam Al Sharou, Zhenhao Li, and Lucia Specia. 2021. Towards a better understanding of noise in natural language processing. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 53–62.

Qiwei Bi, Jian Li, Lifeng Shang, Xin Jiang, Qun Liu, and Hanfang Yang. 2022. Mtrec: Multi-task learning over bert for news recommendation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2663–2669.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, Naman Goyal, Tom Birch, Vitaliy Liptchinsky, Sergey Edunov, Edouard Grave, Michael Auli, and Armand Joulin. 2020. Beyond english-centric multilingual machine translation.

Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini. 2014. *Feedback Control of Dynamic Systems (7th Edition)*. Pearson.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821.

Bohan Li, Yutai Hou, and Wanxiang Che. 2022. Data augmentation approaches in natural language processing: A survey. *AI Open*, 3:71–90.

Sascha Meyen, Frieder Göppert, Helen Alber, Ulrike von Luxburg, and Volker H Franz. 2021. Specialists outperform generalists in ensemble classification. *arXiv preprint arXiv:2107.04381*.

Amit Pandey, Maurício Oliveira, and Robert Moroto. 2018. Model predictive control for gas turbine engines.

Lucas Francisco Amaral Orosco Pellicer, Taynan Maier Ferreira, and Anna Helena Reali Costa. 2023. Data augmentation techniques in natural language processing. *Applied Soft Computing*, 132:109803.

Nils Reimers and Iryna Gurevych. 2019. Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

SambitSekhar. 2017. First quora dataset release: Question pairs.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. *arXiv preprint arXiv:1901.11196*.