**IJCCS (Indonesian Journal of Computing and Cybernetics Systems)**
Vol.16, No.4, October 2022, pp. 367~378
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.72981


# Neural Network Pruning in Unsupervised Aspect Detection based on Aspect Embedding

**Muhammad Haris Maulana*[1], Masayu Leylia Khodra[2]**
[1]Master Program of Informatics, SEEI; ITB, Bandung, Indonesia
[2]School of Electrical Engineering and Informatics, ITB, Bandung, Indonesia
e-mail: *[1]**23519008@std.stei.itb.ac.id** , [2]masayu@staff.stei.itb.ac.id

***Abstrak***

*Sistem deteksi aspek tanpa supervisi dinilai cocok secara strategis untuk mengolah ulasan-ulasan daring yang umumnya berjumlah sangat banyak dan tidak terstruktur. Model Deep Learning berbasis Aspect Embedding merupakan salah satu pendekatan deteksi aspek tanpa supervisi yang dapat mengolah ulasan daring. Namun pendekatan tersebut sensitif terhadap inisialisasi model dan redudansi pada embedding model yang dapat mempengaruhi kinerja model secara signifikan. Dalam penelitian ini, metode Pruning atau pemangkasan bobot digunakan untuk mengurangi redudansi dalam model tersebut dan mencoba menghasilkan model baru dengan kinerja yang serupa atau lebih baik. Terdapat sejumlah eksperimen dan perbandingan kinerja model baru berdasarkan strategi umum pemangkasan bobot dan Lottery Ticket Hypothesis. Hasil penelitian menunjukkan bahwa pemangkasan bobot pada model deteksi aspek tanpa supervisi berbasis Aspect Embedding dapat menghasilkan model baru meskipun sebagian besar bobotnya terpangkas dan masih memiliki kinerja yang hampir setara, Hasil salah satu eksperimen kami menghasilkan model baru dengan 80% koneksi yang dipangkas dengan kinerja model yang setara dengan model sebelumnya. Implementasi pemangkasan bobot dalam penelitian ini belum dapat menghasilkan model baru dengan kenaikan kinerja yang signifikan.*

***Kata kunci****— aspect embedding, deteksi aspek tanpa supervisi, pemangkasan, lottery ticket hypothesis*

***Abstract***

*Aspect detection systems for online reviews, especially based on unsupervised models, are considered better strategically to process online reviews, generally a very large collection of unstructured data. Aspect embedding-based deep learning models are designed for this problem however they still rely on redundant word embedding and they are sensitive to initialization which may have a significant impact on model performance. In this research, a pruning approach is used to reduce the redundancy of deep learning model connections and is expected to produce a model with similar or better performance. This research includes several experiments and comparisons of the results of pruning the model network weights based on the general neural network pruning strategy and the lottery ticket hypothesis. The result of this research is that pruning of the unsupervised aspect detection model, in general, can produce smaller submodels with similar performance even with a significant amount of weights pruned. Our sparse model with 80% of its total weight pruned has a similar performance to the original model. Our current pruning implementation, however, has not been able to produce sparse models with better performance.*

***Keywords—*** *aspect embedding, unsupervised aspect detection, pruning, lottery ticket hypothesis*

*Received February 11th,2022; Revised October 19th, 2022; Accepted October 20th, 2022*

## 1. INTRODUCTION

Online reviews of a product or service often influence the decision-making process of potential customers and product or service providers. These reviews are often provided in large quantities and may require significant human effort to analyze. Automated analysis systems such as which implement aspect-based sentiment analysis can be used to help understand the general opinion in online reviews.

Aspect-Based Sentiment Analysis can analyze texts and generate specific target opinions and sentiments. For example, the sentence "This restaurant has a romantic atmosphere but the waiters are not alert" contains a positive opinion of the atmosphere and a negative opinion of the service [1]. Aspect detection or categorization is one of the subtasks in aspect-based sentiment analysis which aims to extract or identify possible aspects described in a text. For example in the sentence "The sound from the speaker is very clear", the word "Sound" is an aspect expression that determines that the sentence contains the "sound quality" aspect of the "speaker" entity. We can use aspect detection to filter online reviews, which are often provided in large numbers, that contain segments of aspects of interest [2].

Unsupervised learning approaches for aspect detection provide strategic advantages for analyzing online reviews. Supervised learning methods often require a large amount of labeled data for training. Generating labeled data using human annotation may require a large number of resources and time while also still being susceptible to human error. Unsupervised learning models are also more robust to learn from online reviews, which are often short, unstructured, and may contain lexical errors [3].

There are several studies on unsupervised aspect detection (UAD). Most earlier research in unsupervised approaches is influenced by Latent Dirichlet Allocation [4], [5] with several other approaches such as rule-based models [6]. Deep learning models such as aspect-based autoencoder [7], and self-supervised contrastive learning [2] have shown significant results for unsupervised aspect detection tasks. Attention-based Aspect Extraction (ABAE) and Self Supervised Contrastive Learning (SSCL) are both deep learning models that generate aspect-labeled clusters called aspect embedding in the word embedding space. These aspect embeddings are centroids initialized from k-means clustering and further trained to fit each model aspect embedding fitting method.

While these deep learning models perform better than other known UAD models, both models are sensitive to initialization and over-parameterized. Aspect embeddings are initialized as k-means centroids and if a different random state is used in the clustering process then the cluster generation may result in a different initial cluster that may affect model performance. Certain embeddings, such as word embeddings, used in NLP models are often redundant or over-parameterized. Shu and Nakayama [8] showed that a model with 90-99% compressed word embedding achieved no significant performance loss. See et al. [9] showed that the lower layers of Neural Machine Translation (NMT), in particular word embedding, contain a lot of redundancy.

Neural Network Pruning is the task of reducing a neural network size by systematically removing parameters from the existing network while maintaining the network performance [10]. Neural networks, generally deep learning models, tend to have very good performance but require a large amount of computation and memory, resulting in real-world implementation costs. Pruning aims to reduce the size, memory usage, and complexity of a model. Pruning is also expected to reduce dataset overfitting on the network and reduce the impact of over-parameterization. The pruning process generally involves removing some parameter models, typically neural network layers, neurons, or weights, to produce a neural network that is simpler but maintains performance, generally accuracy, on par with the original model.

In this work, we conduct comparative experiments to find the best pruning strategy to implement in the ABAE model with an aspect mapping strategy adapted from the SSCL model

called High-Resolution Mapping (HRSMap). While pruning is often used to reduce the size of a neural network while maintaining accuracy, several studies [10]–[12] suggest that pruning, in earlier stages, may increase model performance which may suggest some weight can be safely pruned to improve the model performance. While the SSCL model itself outperforms the ABAE model, Shi et al. [2] found that the ABAE model that implements HRSMap also performs better than the ABAE model.

## 2. METHODS

### *2.1 Dataset*

Data used in this research are restaurant reviews from New York Citysearch Corpus which are also datasets used in the following research [2], [3], [7]. We also follow the experimental settings from the previous research. 3 categories of aspects will be used: food, service, and ambiance. The distribution of the dataset is shown in Table 1. The training data used in our experiment and previous research are unlabeled from the start. We also used the test data from previous research which are manually labeled with only a single label.

Table 1 Citysearch Dataset Distribution

| Label | Training Data | Test Data |
|---|---|---|
| Food | Unknown | 887 |
| Staff | Unknown | 352 |
| Ambiance | Unknown | 251 |
| Total | 279.862 | 1490 |

### *2.2 Preprocessing*

Preprocessing in this study follows He et al.'s [7] preprocessing process. We remove punctuation symbols, stop words, and words appearing less than 10 times. Word embeddings are trained by word2vec with negative sampling, an embedding size of 200, a window size of 10, and a negative sampling size of 5. For aspect embedding initialization, we follow Shi et al [2] initialization of an aspect size of 30.

### *2.3 Attention-Based Aspect Extraction Model*

He et al. [7] suggest an autoencoder deep learning model that learns aspect embedding by reconstructing attention-weighted sentences. An attention mechanism is used to capture the most relevant word in sentences. Attention-weighted sentences are used as an input for the encoder layer (a dense layer) and we reconstruct the sentence using the decoder output and every aspect embedding. The goal is to reconstruct the attention-weighted sentence with minimum loss by fitting or training the aspect embedding itself. A general overview of the ABAE model structure is shown in Figure 1. There are 3 layers with prunable weights: Sentence Attention Embedding layer (Sen-att), Dense Layer (dense), and Aspect Embedding Layer (asp-emb).

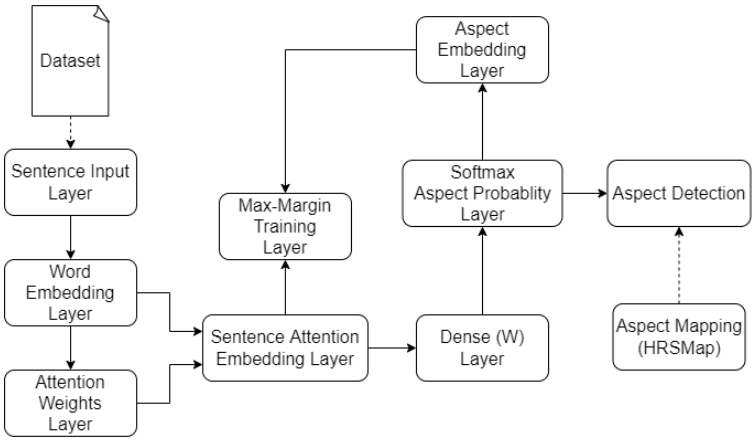


Figure 1 ABAE Model

*2.4 Pruning*

There are 3 types of pruning: layer pruning, neuron pruning, and more commonly, weight pruning. In this work, we focus on weight pruning as there are only 3 trainable layers with prunable weights in the ABAE model. To implement pruning or simulate the removal of weight connections, we added a pruning mask inside the layer with weight. The Pruning mask records the indexes of pruned weights and transforms the value of the pruned indexes into zero. The pruning mask also prevents weight value updates from the backpropagation process. The illustration of the pruning mask implementation is shown in Figure 2.

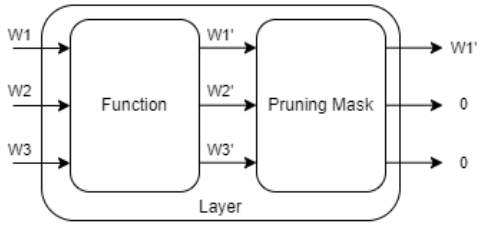


Figure 2 Pruning Mask

*2.5 Experiment Scenario*

There are 5 experiment scenarios based on pruning strategies adapted from several studies [10], [11], [13].

*2.5.1 One-Shot Pruning*

A single-step pruning process in which the model is first trained until convergence and removes all connections whose weight is lower than a threshold determined by the pruning scenario.

*2.5.2 Prune & Fine-tuning*

The iterative pruning [13] process in which the model is first trained until convergence, removes all connections whose weight is lower than a threshold determined by the pruning scenario, fine-tunes or retrains the sparse model, and compares the result with the former model. Iterate, prune, retrain and compare as long as there is not a significant drop in performance. The fine-tuning process is a full model training with the training dataset.

*2.5.3 Prune & Reset*

An alternative to the single-step pruning process is based on the lottery ticket hypothesis [11]. First, save the initialized parameters of the model, the model is then trained until convergence, remove all connections whose weight is lower than a threshold determined by the pruning scenario, and reset all remaining connection weight values to their original value at initialization. By resetting the remaining weights, we are trying to find a *winning ticket*, a subnetwork or sparse model that when trained in isolation reaches a level of performance comparable to the original.

*2.5.4 One-shot Prune, Reset, & Fine-tuning*

Iterative pruning process based on lottery ticket hypothesis. Several NLP research [14], [15] on lottery-based pruning also retrain their model at a certain step. We simply extend the previous experiment scenario by adding a fine-tuning process, that is re-training the pruned model with the training dataset.

*2.5.5 One-shot Prune, Reset, & Fine-tuning* with Random Initialization

The scenario involves the extension of the previous scenario by modifying the reset step by reinitializing the weight values randomly after the model is pruned. Random initialization, based on lottery ticket research [11], may increase the efficiency of finding the winning tickets.

*2.6 Experiment Model*

We used experiment results in previous research [2], [7] for comparison with our pruned models. To compare model performance after the pruning experiment, it is necessary to produce a baseline model to be used as a reference for comparing the results between experiments. The baseline model is built by reproducing the ABAE model and adapting HRSMapping which has better overall performance than the ABAE Model [2]. Table 2 is a performance comparison between the previous models [2], [7], and our baseline model. We used F-measure to calculate model accuracy by calculating Precision (P), Recall (R), and the F1 Score system (F1). Although in this work, model parameters were set the same as in the previous work, the Baseline model has not reached the same performance as the reproduction target model due to difficulty recreating similar environment experiments. Nevertheless, the Baseline model still outperforms the ABAE Model and is used as our experiment baseline model.

Table 2 Baseline Model Comparison

| Model | Food | | | Staff | | | Ambiance | | | Overall | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| ABAE | 95,3 | 86,4 | 87,2 | 80,2 | 72,8 | 75,7 | 81,5 | 69,8 | 74,0 | 89,4 | 73,0 | 79,6 |
| ABAE + HRSMap | 93,0 | 88,8 | 90,9 | 85,8 | 75,3 | 80,2 | 67,4 | 89,6 | 76,9 | 87,0 | 85,8 | 86,0 |
| Baseline | 87,9 | 90,3 | 89,1 | 75,2 | 66,5 | 70,6 | 72,4 | 68,9 | 70,6 | 82,3 | 81,1 | 81,6 |

*2.7 Experiment Design*

We used the same F1 Scoring system for comparison with previous research [2], [7]. We define a significant drop in performance if the pruned model performance reaches more than a -5% difference from the baseline model. If during iterative pruning the experiment reaches a significant drop in performance, the result will be dropped and no more experiments in the pruning scenario will be conducted. There are two pruning scenarios for this experiment: Magnitude-based pruning and Percentage-based pruning.

We decide not to measure model training time or inference speed. While one-shot pruning can be done in relatively a short time, we found that fine-tuning or retraining our sparse models took a similar amount of time as the first model training time. This may have been caused by our adaptation to the experiment environment and settings from previous research [7]. The retraining process does not exclude pruned weights from retraining because we have not been able to adjust the training process with our pruning masks. The retraining process treats pruned connections as normal connections and recalculates their weight values, even if the value of pruned weights will always be zero.

*2.7.1 Pruning Scenario*

Magnitude-based pruning removes all connections whose weight is lower than the maximum weight value times the magnitude. For example in a layer with many connections with weights ranging from -5 to 5, if we designate the magnitude of pruning as 10% then all connections whose absolute value of weight is lower than 10% of the highest absolute value of weight (which is 5x10%)

Percentage pruning sorts all weights based on their absolute value and then removes all connections whose weight value is the bottom of a certain percentage. For example, in a layer with certain weighted connections, if we designate the percentage of pruning as 10% then all connections whose absolute value of weight is in the bottom 10% of all weight.

## 3. RESULTS AND DISCUSSION

*3.1 One-Shot Pruning*

The One-Shot Pruning approach was able to produce a similar model with insignificant weight changes, corresponding with the pruning studies' conclusion observed by Blalock et

al.[10]. However, none of the experiments managed to significantly improve model performance after pruning, only certain aspects have improved their performance in precision, recall, or F1 Score at a certain threshold. The performance of a model with 10-80% of the total weight pruned is close to the baseline model, with a maximum difference of around 1-3% in the F1 Score. Removing 90% of the total weight resulted in a significant drop in performance compared to the baseline model. The comparison between the baseline model and sparse models pruned by one-shot pruning is shown in . The F1 scores of the sparse model remain close to the baseline model score, with each iteration of pruning increasing the difference between the score, however, none of the sparse models were able to achieve better performance.

Table 3. The F1 scores of the sparse model remain close to the baseline model score, with each iteration of pruning increasing the difference between the score, however, none of the sparse models were able to achieve better performance.

Table 3 One-shot Pruning Experiment Results

| Baseline Model | | | | | |
|---|---|---|---|---|---|
| Label | Precision | Recall | F1-Score | Layer | Weights |
| Food | 0,879 | 0,903 | 0,891 | Sen-att | 40.000 |
| Staff | 0,752 | 0,665 | 0,706 | Dense | 6.000 |
| Ambiance | 0,724 | 0,689 | 0,706 | Asp-emb | 6.000 |
| Average | **0,823** | **0,811** | **0,816** | **Total** | **52.000** |
| Label | Precision | Recall | F1-Score | Pruned Weights | |
| 5% Magnitude Pruning | | | | | |
| Food | 0,877 | 0,900 | 0,888 | Sen-att | 7.217 |
| Staff | 0,744 | 0,653 | 0,696 | Dense | 936 |
| Ambiance | 0,715 | 0,689 | 0,702 | Asp-emb | 973 |
| Average | **0,818** | **0,806** | **0,811** | **Total** | **9.126** |
| 10% Magnitude Pruning | | | | | |
| Food | 0,878 | 0,892 | 0,885 | Sen-att | 14.160 |
| Staff | 0,730 | 0,659 | 0,693 | Dense | 1.851 |
| Ambiance | 0,713 | 0,681 | 0,697 | Asp-emb | 1.884 |
| Average | **0,815** | **0,801** | **0,808** | **Total** | **17.895** |
| 25% Magnitude Pruning | | | | | |
| Food | 0,930 | 0,809 | 0,866 | Sen-att | 29.603 |
| Staff | 0,775 | 0,685 | 0,727 | Dense | 4.043 |
| Ambiance | 0,558 | 0,785 | 0,652 | Asp-emb | 4.151 |
| Average | **0,831** | **0,776** | **0,797** | **Total** | **37.797** |
| 10% Total Weight Pruning | | | | | |
| Food | 0,879 | 0,908 | 0,893 | Sen-att | 4.000 |
| Staff | 0,761 | 0,659 | 0,706 | Dense | 600 |
| Ambiance | 0,721 | 0,689 | 0,705 | Asp-emb | 600 |
| Average | **0,824** | **0,812** | **0,817** | **Total** | **5.200** |
| 80% Total Weight Pruning | | | | | |
| Food | 0,936 | 0,803 | 0,864 | Sen-att | 32.000 |
| Staff | 0,748 | 0,682 | 0,713 | Dense | 4.800 |
| Ambiance | 0,555 | 0,661 | 0,604 | Asp-emb | 4.800 |
| Average | **0,827** | **0,750** | **0,785** | **Total** | **41.600** |
| 90% Total Weight Pruning | | | | | |
| Food | 0,828 | 0,759 | 0,792 | Sen-att | 36.000 |
| Staff | 0,607 | 0,668 | 0,636 | Dense | 5.400 |
| Ambiance | 0,551 | 0,558 | 0,554 | Asp-emb | 5.400 |
| Average | **0,729** | **0,703** | **0,715** | **Total** | **46.800** |

*3.2 Prune & Fine-tuning*

The iterative pruning approach based on pruning research by Han et al. [13] was able to produce a similar model with insignificant weight changes, corresponding with the pruning studies' conclusion observed by Blalock et al.[10]. However, none of the experiments managed to significantly improve model performance after pruning. Overall, Compared to One-shot pruning, sparse models pruned by iterative pruning have higher performance loss. The fine-tuning process which retrains the model with a full training dataset may have resulted in higher performance loss during testing with test data. The full result of the sparse model pruned with iterative pruned is shown in Table 4. Similar results from the previous experiment are also shown, with the F1 Score of the sparse model being close but below the baseline model, and each iteration of pruning and fine-tuning increases the distance. Fine-tuning may not always increase model performance as we see in Tables 3 and 4, a sparse model pruned with the one-shot method with 25% magnitude pruning achieved better performance than its fine-tuned sparse model counterpart.

Table 4 Prune & Fine-tuning Experiment Result

| Baseline Model | | | | | |
|---|---|---|---|---|---|
| Label | Precision | Recall | F1-Score | Layer | Weights |
| Food | 0,879 | 0,903 | 0,891 | Sen-att | 40.000 |
| Staff | 0,752 | 0,665 | 0,706 | Dense | 6.000 |
| Ambiance | 0,724 | 0,689 | 0,706 | Asp-emb | 6.000 |
| Average | **0,823** | **0,811** | **0,816** | **Total** | **52.000** |
| Label | Precision | Recall | F1-Score | Pruned Weights | |
| 10% Magnitude Pruning Retrain 3 time | | | | | |
| Food | 0,875 | 0,840 | 0,857 | Sen-att | 19.510 |
| Staff | 0,586 | 0,705 | 0,640 | Dense | 2.806 |
| Ambiance | 0,784 | 0,534 | 0,635 | Asp-emb | 1.712 |
| Average | **0,792** | **0,756** | **0,769** | **Total** | **24.028** |
| 25% Magnitude Pruning Retrain 2 times | | | | | |
| Food | 0,876 | 0,814 | 0,844 | Sen-att | 30.969 |
| Staff | 0,668 | 0,705 | 0,686 | Dense | 4.900 |
| Ambiance | 0,588 | 0,614 | 0,600 | Asp-emb | 4.207 |
| Average | **0,779** | **0,754** | **0,766** | **Total** | **40.076** |
| 20% Total Weight Pruning Retrain 1 time | | | | | |
| Food | 0,873 | 0,887 | 0,880 | Sen-att | 8.000 |
| Staff | 0,699 | 0,679 | 0,689 | Dense | 1.200 |
| Ambiance | 0,763 | 0,629 | 0,690 | Asp-emb | 1.200 |
| Average | **0,814** | **0,795** | **0,803** | **Total** | **10.400** |
| 20% Total Weight Pruning Retrain 3 times | | | | | |
| Food | 0,872 | 0,823 | 0,847 | Sen-att | 24.000 |
| Staff | 0,564 | 0,716 | 0,631 | Dense | 3.600 |
| Ambiance | 0,767 | 0,498 | 0,604 | Asp-emb | 3.600 |
| Average | **0,782** | **0,743** | **0,755** | **Total** | **31.200** |
| 80% Total Weight Pruning Retrain 1 time | | | | | |
| Food | 0,916 | 0,822 | 0,866 | Sen-att | 32.000 |
| Staff | 0,711 | 0,705 | 0,708 | Dense | 4.800 |
| Ambiance | 0,596 | 0,705 | 0,646 | Asp-emb | 4.800 |
| Average | **0,813** | **0,774** | **0,792** | **Total** | **41.600** |

*3.3 Prune & Reset*

The prune & reset or finding *winning tickets* in one cycle approach experienced a very significant decrease in performance. The implementation of trimming the weights for each lower threshold and percentage of the overall weight reduces performance significantly. This may be due to the pruning experiment and uncertain conditions caused by the model structure not being able to produce a winning ticket condition.

The result of the experiment is shown in Table 5. The sparse model performance drops significantly despite the pruning process done in a single iteration or more. The performance drop may have been caused during the reset process. When the surviving weights are reset to initialization, it also causes the Aspect embedding position in the embedding space to reset to its initial k-means centroids which may cause overgeneralization. However, during our second experiment with percentage pruning with freezing the aspect embedding layer, the sparse model still performed worse over iterations. This experiment may signify the problem of finding winning tickets, that is finding the initial initialization values that work well with the sparse models, in our current experiment setting is too broad as there could have been several parameters that may affect the model performance.

Table 5 Prune & Reset Experiment Result

| Baseline Model | | | | | |
|---|---|---|---|---|---|
| Label | Precision | Recall | F1-Score | Layer | Weights |
| Food | 0,879 | 0,903 | 0,891 | Sen-att | 40.000 |
| Staff | 0,752 | 0,665 | 0,706 | Dense | 6.000 |
| Ambiance | 0,724 | 0,689 | 0,706 | Asp-emb | 6.000 |
| Average | **0,823** | **0,811** | **0,816** | **Total** | **52.000** |
| Label | Precision | Recall | F1-Score | Pruned Weights | |
| 10% Magnitude Pruning | | | | | |
| Food | 0,647 | 0,253 | 0,363 | Sen-att | 14.160 |
| Staff | 0,252 | 0,088 | 0,131 | Dense | 1.851 |
| Ambiance | 0,129 | 0,498 | 0,205 | Asp-emb | 1.884 |
| Average | **0,467** | **0,225** | **0,282** | **Total** | **17.895** |
| 40% Magnitude Pruning | | | | | |
| Food | 0,468 | 0,343 | 0,396 | Sen-att | 36.727 |
| Staff | 0,214 | 0,381 | 0,274 | Dense | 5.304 |
| Ambiance | 0,028 | 0,020 | 0,023 | Asp-emb | 5.417 |
| Average | **0,334** | **0,297** | **0,304** | **Total** | **47.448** |
| 10% Total Weight Pruning | | | | | |
| Food | 0,640 | 0,878 | 0,740 | Sen-att | 4.000 |
| Staff | 0,606 | 0,162 | 0,256 | Dense | 600 |
| Ambiance | 0,341 | 0,239 | 0,281 | Asp-emb | 0 |
| Average | **0,581** | **0,601** | **0,548** | **Total** | **4.600** |
| 60% Total Weight Pruning | | | | | |
| Food | 0,421 | 0,336 | 0,374 | Sen-att | 24.000 |
| Staff | 0,115 | 0,114 | 0,114 | Dense | 3.600 |
| Ambiance | 0,482 | 0,159 | 0,159 | Asp-emb | 0 |
| Average | **0,359** | **0,254** | **0,254** | **Total** | **27.600** |

*3.4 One-Shot Prune, Reset, & Fine-tuning*

The combination of iterative pruning and resetting the unpruned weights to their initial weight based on the lottery ticket hypothesis [11], [14] produced sparse models with minor changes in performance similar to normal Iterative pruning [13]. None of the experiments however managed to significantly improve model performance after pruning. Like the previous iterative pruning, iterations of fine-tuning or retraining may contribute to the performance loss of the sparse models. A sparse model with 80% of its total weights pruned with two iterations of fine-tuning has better performance than a sparse model with more fine-tuning iterations. The Comparison between the baseline model and sparse models is shown in Table 6. Despite extending the previous experiment method, the sparse model performance is comparable with the iterative pruning done in the 3.2 Prune & Fine-tuning.

Table 6 One-shot Prune, Reset & Fine-tuning Experiment Result

| Baseline Model* | | | | | |
|---|---|---|---|---|---|
| Label | Precision | Recall | F1-Score | Layer | Weight |
| Food | 0.879 | 0,903 | 0,891 | Attention | 40.000 |
| Staff | 0,752 | 0,665 | 0,706 | Dense | 6.000 |
| Ambiance | 0,724 | 0,689 | 0,706 | Asp_emb | 6.000 |
| Average | **0,823** | **0,811** | **0,816** | **Total** | **52.000** |
| Label | Precision | Recall | F1-Score | Pruned Weights | |
| 10% Magnitude Pruning & Retrain 4 iterations | | | | | |
| Food | 0,889 | 0,832 | 0,870 | Attention | 17.857 |
| Staff | 0,707 | 0,528 | 0,605 | Dense | 3.666 |
| Ambiance | 0,680 | 0,769 | 0,721 | Asp_emb | 1.814 |
| Average | **0,811** | **0,750** | **0,776** | **Total** | **23.337** |
| 20% Percentage Pruning & Retrain 1 iteration | | | | | |
| Food | 0,899 | 0,843 | 0,870 | Attention | 8.000 |
| Staff | 0,615 | 0,722 | 0,664 | Dense | 1.200 |
| Ambiance | 0,732 | 0,685 | 0,708 | Asp_emb | 1.200 |
| Average | **0,804** | **0,788** | **0,794** | **Total** | **10.400** |
| 20% Percentage Pruning & Retrain 3 iterations | | | | | |
| Food | 0,850 | 0,820 | 0,835 | Attention | 24.000 |
| Staff | 0,780 | 0,554 | 0,648 | Dense | 3.600 |
| Ambiance | 0,490 | 0,745 | 0,591 | Asp_emb | 3.600 |
| Average | **0,773** | **0,744** | **0,749** | **Total** | **31.200** |
| 80% Percentage Pruning & Retrain 2 iterations | | | | | |
| Food | 0,904 | 0,858 | 0,880 | Attention | 32.000 |
| Staff | 0,792 | 0,628 | 0,700 | Dense | 4.800 |
| Ambiance | 0,564 | 0,757 | 0,646 | Asp_emb | 4.800 |
| Average | **0,820** | **0,787** | **0,798** | **Total** | **41.600** |

*3.5 One-Shot Prune, Reset, & Fine-tuning* (OPRF) with Random Initialization

We expanded the previous experiment scenario by randomly initializing the surviving weights based on previous research [11], [14], [15]. We produced new sparse models with varying performance results, mostly minor changes in performance. There is one sparse model with better performance than the baseline model, while the rest of the sparse model produced has a similar result to previous iterative pruning methods. The best sparse model achieved an F1 Score of 0,824 or a 1% improvement over the baseline model, which is still a minor improvement. The sparse

model may be one of the winning tickets that fit with the combination of pruning parameters, initial initialization parameters, and random surviving weight initialization. The Comparison between the baseline model and the best sparse models during this experiment is shown in Table 7.

Table 7 OPRF with Random Initialization Experiment Result

| Baseline Model* | | | | | |
|---|---|---|---|---|---|
| Label | Precision | Recall | F1-Score | Layer | Weights |
| Food | 0,879 | 0,903 | 0,891 | Attention | 40.000 |
| Staff | 0,752 | 0,665 | 0,706 | Dense | 6.000 |
| Ambiance | 0,724 | 0,689 | 0,706 | Asp_emb | 6.000 |
| Average | **0,823** | **0,811** | **0,816** | **Total** | **52.000** |
| Label | Precision | Recall | F1-Score | Pruned Weights | |
| 10% Percentage Pruning & Retrain 1st iteration | | | | | |
| Food | 0,909 | 0,776 | 0,837 | Attention | 4.000 |
| Staff | 0,826 | 0,486 | 0,612 | Dense | 600 |
| Ambiance | 0,793 | 0,367 | 0,501 | Asp_emb | 600 |
| Average | **0,870** | **0,638** | **0,727** | **Total** | **5.200** |
| 10% Percentage Pruning & Retrain 2nd iteration | | | | | |
| Food | 0,868 | 0,931 | 0,898 | Attention | 8.000 |
| Staff | 0,915 | 0,582 | 0,712 | Dense | 1.200 |
| Ambiance | 0,685 | 0,761 | 0,721 | Asp_emb | 1.200 |
| Average | **0,848** | **0,820** | **0,824** | **Total** | **10.400** |
| 10% Percentage Pruning & Retrain 3rd iteration | | | | | |
| Food | 0,865 | 0,744 | 0,800 | Attention | 12.000 |
| Staff | 0,483 | 0,702 | 0,572 | Dense | 1.800 |
| Ambiance | 0,844 | 0,518 | 0,642 | Asp_emb | 1.800 |
| Average | **0,771** | **0,696** | **0,720** | **Total** | **15.600** |
| 10% Percentage Pruning & Retrain 4th iteration | | | | | |
| Food | 0,805 | 0,926 | 0,861 | Attention | 16.000 |
| Staff | 0,797 | 0,537 | 0,642 | Dense | 2.400 |
| Ambiance | 0,831 | 0,586 | 0,687 | Asp_emb | 2.400 |
| Average | **0,807** | **0,777** | **0,780** | **Total** | **20.800** |

## 4. CONCLUSIONS

Our current implementations of pruning techniques and experiments showed that we could prune a significant amount of weights from attention-based aspect extraction models based on previous research [7] and still achieved similar performance with a mostly minor decrease in performance. However, our current pruning implementation has not been able to reproduce sparse models with a major performance improvement. Experiment results in Tables 8 and 12 have shown that sparse models with 80% of total weight pruned achieved F1 Scores of 0,794 and 0,798, only a 2-3% difference from the baseline model with an F1 Score of 0,816. Our best sparse model shown in Table 13 managed to outperform the baseline model with a merely minor performance improvement of 1%. The current implementation of the lottery ticket hypothesis in our research has not been able to reproduce winning tickets favorably which may be caused by several unknown factors that render the sparse model difficult to find the *winning ticket*.

The fine-tuning process after pruning in our research often results in higher performance loss after several iterations. Based on our current experiment results, it is better to prune more weights and retrain in fewer iterations or even skip retraining. For example, the performance of a

model whose total weight was trimmed by 80% and retrained once is better than the performance of a model whose total weight was trimmed by 20% and retrained 4 times iteratively.

Suggestions for further research are the implementation of a more selective pruning method and finding winning tickets more effectively. As all layers are pruned equally determined by a certain threshold per layer, a more selective pruning approach by considering factors such as the possibility that the dense layer is too important to prune may result in a better model. Investigating possible alternatives for the fine-tuning process may also improve the model performance to reduce the impact of biases from training data. There are also several pruning methods not researched for this model, especially methods related to finding winning tickets. Research of the aforementioned conditions and limitations and also alternative pruning methods may result in finding a better methodology to create sparse models with comparable or even better performance.

## REFERENCES

[1] M. Pontiki, D. Galanis, J. Pavlopoulos, H. Papageorgiou, I. Androutsopoulos, and S. Manandhar, "SemEval-2014 Task 4: Aspect Based Sentiment Analysis," p. 9, 2014.

[2] T. Shi, L. Li, P. Wang, and C. K. Reddy, "A Simple and Effective Self-Supervised Contrastive Learning Framework for Aspect Detection," p. 13, Aug. 2020.

[3] S. Brody and N. Elhadad, "An Unsupervised Aspect-Sentiment Model for Online Reviews," *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics.*, 2010.

[4] A. García-Pablos, M. Cuadros, and G. Rigau, "W2VLDA: Almost unsupervised system for Aspect Based Sentiment Analysis," *Expert Systems with Applications*, vol. 91, pp. 127–137, Jan. 2018, doi: 10.1016/j.eswa.2017.08.049.

[5] A. Mukherjee and B. Liu, "Aspect Extraction through Semi-Supervised Modeling," p. 10, 2012.

[6] Q. Liu, B. Liu, Y. Zhang, D. S. Kim, and Z. Gao, "Improving Opinion Aspect Extraction Using Semantic Similarity and Aspect Associations," p. 7, 2016.

[7] R. He, W. S. Lee, H. T. Ng, and D. Dahlmeier, "An Unsupervised Neural Attention Model for Aspect Extraction," presented at the Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, Canada, 2017. doi: 10.18653/v1/P17-1036.

[8] R. Shu and H. Nakayama, "Compressing Word Embeddings via Deep Compositional Code Learning," *arXiv:1711.01068 [cs]*, Nov. 2017, Accessed: Jun. 24, 2021. [Online]. Available: http://arxiv.org/abs/1711.01068

[9] A. See, M.-T. Luong, and C. D. Manning, "Compression of Neural Machine Translation Models via Pruning," in *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, Berlin, Germany, Aug. 2016, pp. 291–301. doi: 10.18653/v1/K16-1029.

[10] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the State of Neural Network Pruning?," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 129–146, Mar. 2020.

[11] J. Frankle and M. Carbin, "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks," *arXiv:1803.03635 [cs]*, Mar. 2019, Accessed: Jun. 02, 2021. [Online]. Available: http://arxiv.org/abs/1803.03635

[12] B. Bartoldson, A. Morcos, A. Barbu, and G. Erlebacher, "The Generalization-Stability Tradeoff In Neural Network Pruning," in *Advances in Neural Information Processing Systems*, 2020, vol. 33, pp. 20852–20864. Accessed: Aug. 31, 2022. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/ef2ee09ea9551de88bc11fd7eeea93b0-Abstract.html

[13] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, Cambridge, MA, USA, December 2015, pp. 1135–1143.

[14] H. Yu, S. Edunov, Y. Tian, and A. S. Morcos, "Playing the lottery with rewards and multiple languages: lottery tickets in RL and NLP," *arXiv:1906.02768 [cs, stat]*, Feb. 2020, Accessed: Sep. 21, 2021. [Online]. Available: http://arxiv.org/abs/1906.02768

[15] M. Behnke and K. Heafield, "Losing Heads in the Lottery: Pruning Transformer Attention in Neural Machine Translation," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online, 2020, pp. 2664–2674. doi: 10.18653/v1/2020.emnlp-main.211.