Deep Learning for Audio and Music (ECS7013P) Coursework Report

Chris Winnard - 210743674

April 2022

1 Introduction

I have worked on two problems for speech: multi-class emotion classification, and binary gender classification.

I originally proposed focusing on multi-label emotion and multi-class genre classification of music in the Emotify dataset [1]. However, there were limited resources available to help with this. For example, to my knowledge there are no existing multi-label emotion classifiers for audio which are available to use as a starting point. Deviating to multi-class emotion classification was considered, however no appropriately-labelled datasets. I thus deviated further to multi-class emotion and binary gender classification of speech.

I solved the emotion classification problem using an existing Jupyter notebook which was designed to work with the dataset used. The workflow involves applying noise (for robustness), creating mel spectrograms, and then training/testing a CNN classifier. I improved the network performance through hyperparameter tuning.

I then created my own CNN for gender classification. This was simpler than the emotion classifier, and the preprocessing steps did not include adding artificial noise. Nevertheless, it was able to achieve better performance than the emotion classifier. This is at least in part because binary gender classification is a simpler problem.

2 Datasets

The main dataset used is the subset of the RAVDESS dataset consisting of audio-only speech recordings [2]. This subset composed of a total of 1440 speech recordings contributed evenly by 24 speakers, with an even gender split. Each recording features the statement "kids are talking by the door", or "dogs are sitting by the door". These are spoken with different emotions: neutral; calm; happy; sad; angry; fearful; disgusted; and surprised. Non-neutral emotions could be strong or normal. Up to two recordings of the same statement/emotion/intensity combination could be included for a given actor. The salient point is that there are eight mutually exclusive emotion labels and two gender labels used.

For emotion classification, noise from the UrbanSound8K dataset [3] is used. As in the original workflow, only the fold1 subset of UrbanSound8K was needed. This contains 873 recordings of sounds heard in a city.

3 Methodology

3.1 Emotion Classification

I have re-implemented part of the workflow described in [4], which is open-source [5]. This is shown in Figure 1, with Figure 2 showing the network. The workflow takes in the speech-only audio from the RAVDESS dataset and duplicated it to create five versions: the original audio; audio with white noise applied; audio with urban noise applied; audio with both noise types applied; and an aggregation of the other four. Then, mel spectrograms are produced, and for each dataset these are split between training data (60% per dataset); validation data (20%); and test data. A CNN is then trained and tested, using each dataset separately. The original Jupyter notebook includes code for music recommendation, which I removed before use.

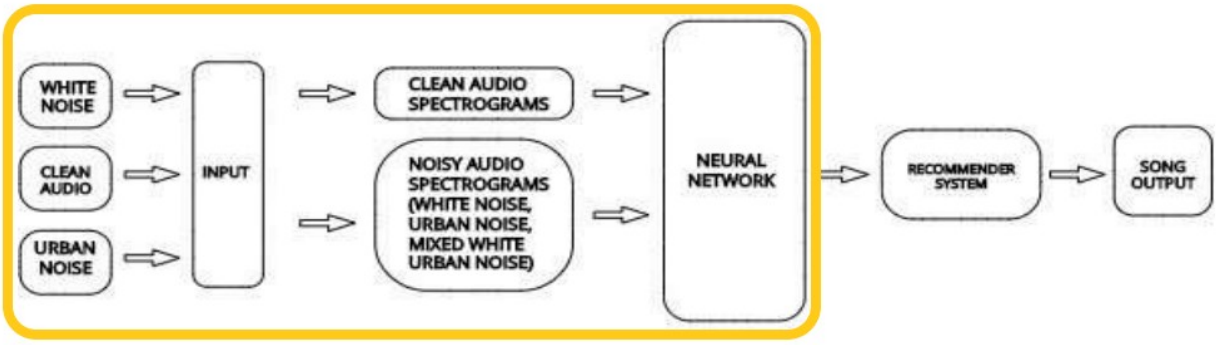

Figure 1: The workflow which was used as a starting point for multi-class emotion classification. Only the parts in the orange box were used for this assignment, as the later stages are used for music recommendation [4].

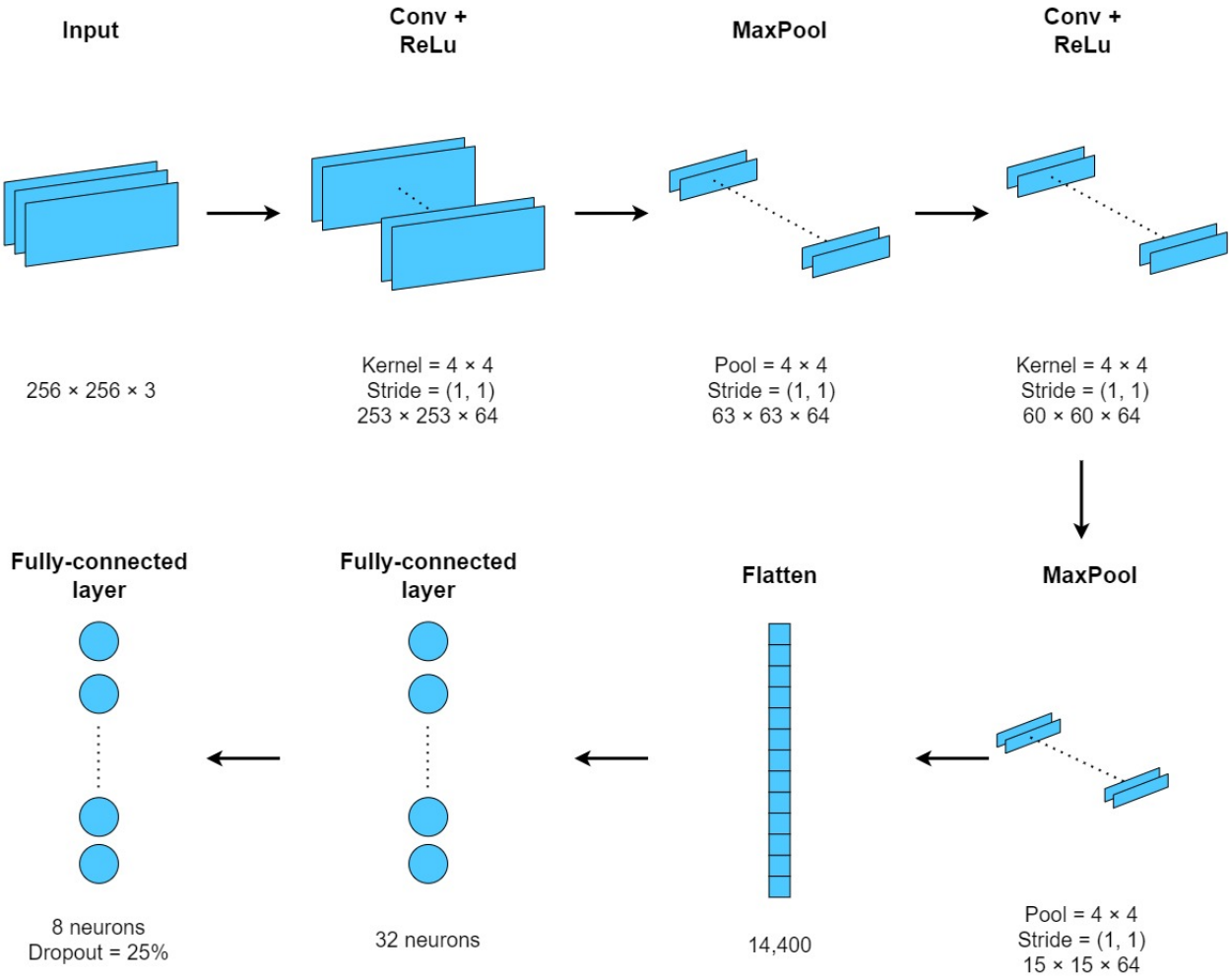

Figure 2: The CNN uses batch sizes of 50. It runs up to 200 epochs of 51 steps, with 18 steps for final validation. It stops early if the validation loss does not improve by ≥ 0.03 for three epochs.

Datasets with different noise augmentations are used because applying noise may benefit robustness and help to prevent overfitting, but it is not obvious which noise type (if any) to apply, or if it is best to apply noise

to all files. Hence, it is sensible to use multiple versions of the same original dataset, and test which leads to optimal results.

Table 1 shows the result metrics from implementation. These are similar to those from [4], including the fact that the network is most effective when run on the aggregation dataset. For example, this dataset yields the highest F1-score both within my implementation (0.829), and within the reference work (0.792). Overall, the fact that the test results are of a high quality and similar to the reference results shows that I have implemented the model correctly. Minor differences between my results and the reference results can be explained by the randomness of set splitting.

Data	Loss	Accuracy	AUC	F1-score
Clean	0.710	0.944	0.963	0.744
White noise applied	0.866	0.939	0.943	0.720
Urban noise applied	1.013	0.924	0.924	0.632
White + urban noise applied	0.846	0.933	0.951	0.676
Aggregation	0.607	0.959	0.972	0.829
Aggregation (reference)	-	0.953	-	0.792

Table 1: The test results from implementing the CNN for the five versions of the dataset. For reference, the results for the aggregation dataset from the original work with the network have been included, as these were the highest-quality results found during that work [4].

3.1.1 Network Improvement

As the network works best with the aggregated dataset, I used this during tuning. I used a trial-and-error method to optimise the dropout rate, and then the learning rate. I found the optimal combination to be a 25% dropout rate, and a learning rate of 0.0022 (10% greater than the original rate). I believe that increasing the rate helps to prevent the system from being trapped in local minima. The tuned model has 529,832 trainable parameters. Table 2 and Figure 3 show its results.

Although the improvements are modest, through hyperparameter tuning I was able to improve the performance of the model in each metric.

Metric	Original model	Tuned model	Tuned model - original model
Loss	0.607	0.470	-0.137
Accuracy	0.959	0.966	0.007
AUC	0.972	0.982	0.010
Precision	0.869	0.911	0.042
Recall	0.793	0.807	0.014
F1-score	0.829	0.856	0.027

Table 2: Test results for the aggregate dataset.

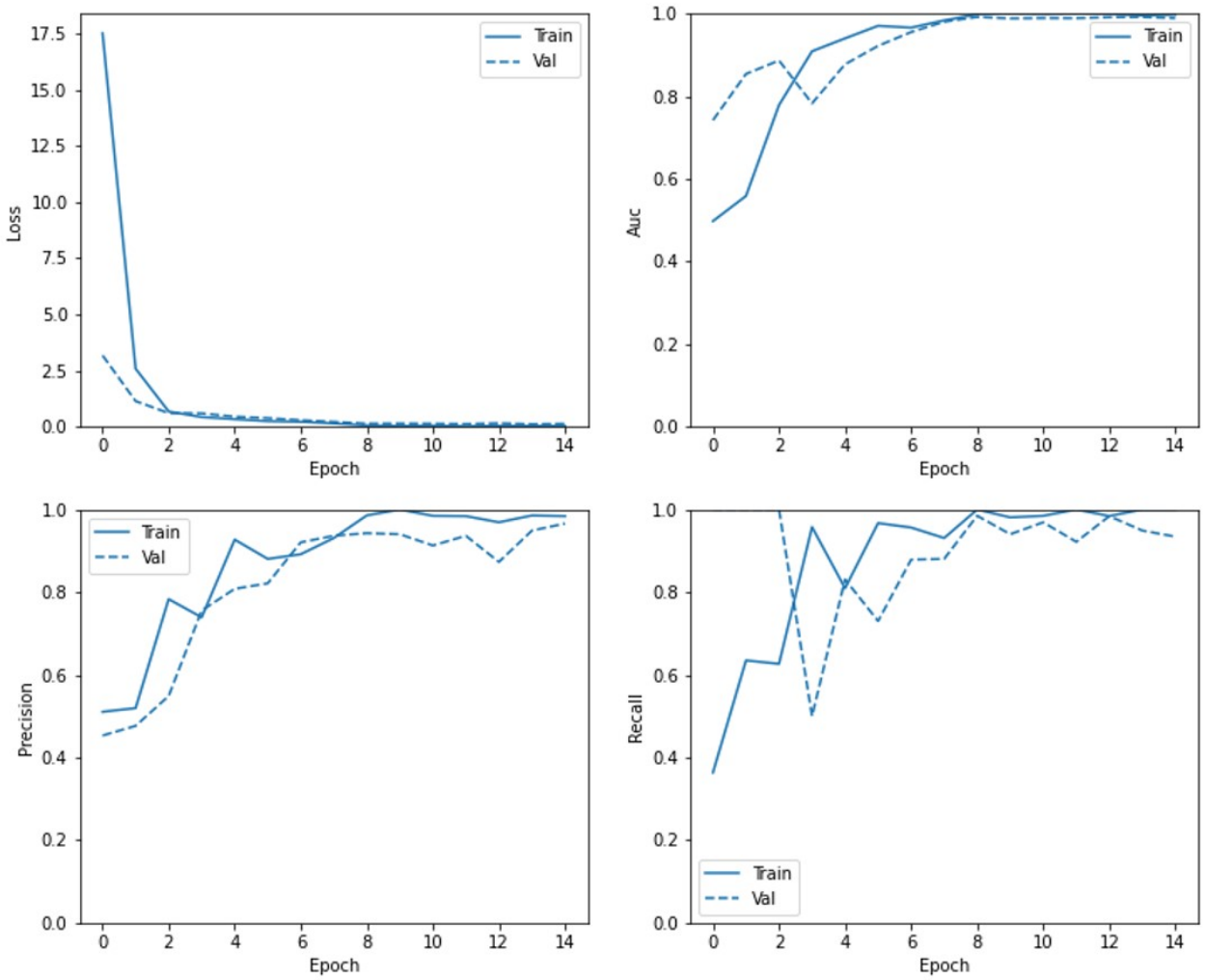

Figure 3: Performance of the tuned model training on the aggregated dataset.

3.1.2 Case Study

There is a bug which causes failure when the code is used to make a prediction for a single audio file. Due to time constraints, I have been unable to rectify this.

3.2 Speaker Gender Classification

For this task I designed a simple CNN, shown in Figure 4.

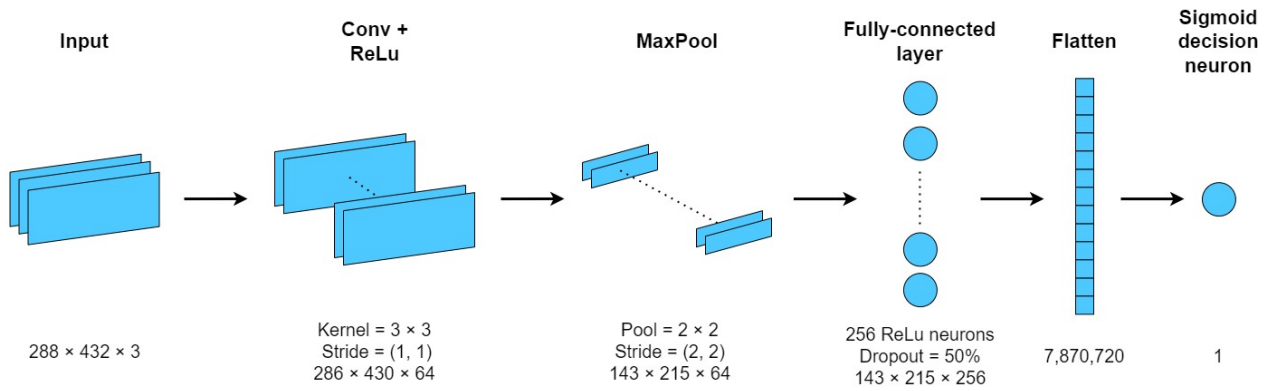

Figure 4: The CNN for gender classification, after hyperparameter tuning.

I chose to work with mel spectrograms as this can be more effective than working with raw audio for gender classification [6]. I took code snippets and inspiration from [5], [7], [8] for preprocessing/the network/metric plotting, but I designed the architecture.

I chose to use one convolutional/ReLu layer, one MaxPool layer, and a fully-connected layer with dropout as this has been shown to be optimal for binary image classification [9]. This is followed by a flatten layer to convert the data into a vector format, and a sigmoid decision neuron.

I originally set the dropout retention probability, p, as 0.5, as this is known to be an effective value for fully-connected layers dealing with abstract representations of the data [10], [11]. I used the Adam optimisation algorithm, which is widely considered to be optimal for such applications [12], [13].

Informed by [6] I split the recordings into three sets: training (12 speakers), validation (six speakers), and test (six speakers). Each set had an even gender split. This was following [6]. Due to time constraints, I did not use cross-validation.

3.2.1 Hyperparameter Tuning

Informed by [8], I used 15 epochs, each with eight steps, and this was found to lead to effective convergence with efficient fitting (less than 15 minutes on Google Colab).

Through trial-and-error guided by [14]–[17], I tuned the other hyperparameters. The optimal combination is: a learning rate of 0.001; a batch size of 16; 256 neurons in the fully-connected layer; and $p = 0.5$. The tuned network has 7,889,153 trainable parameters.

See Figure 5 and Table 3 for the tuned network's performance during training and testing respectively.

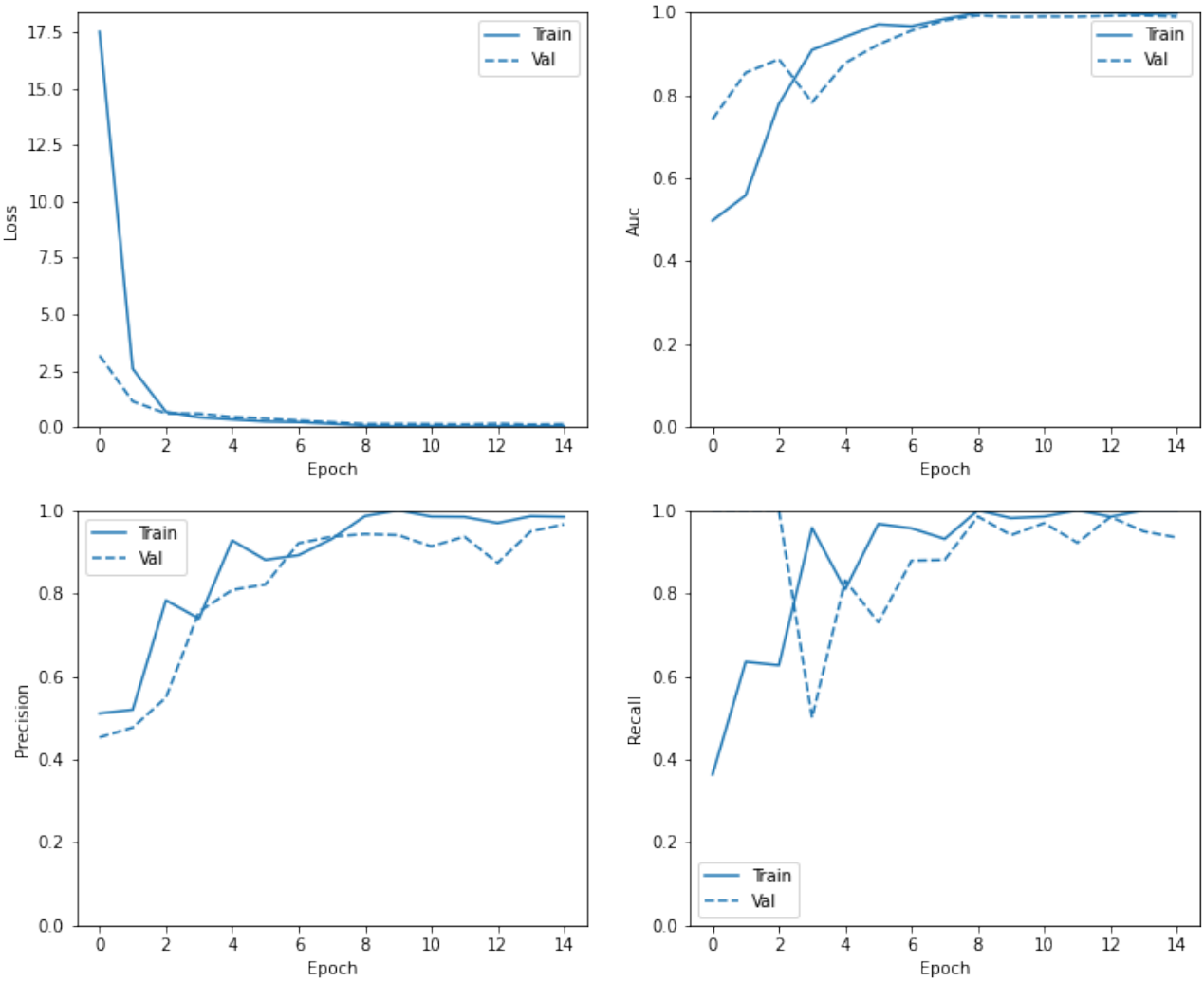

Figure 5: Performance of my tuned model for gender classification as it trains.

Metric	Test value
Loss	0.138
Accuracy	0.944
AUC	0.990
Precision	0.965
Recall	0.922
F1-score	0.943

Table 3: The test results of my tuned model for gender classification.

3.2.2 Case Study

I used a clip from George W Bush Public Domain Audio Archive [18], as this contains copyleft speech recordings which are ≤ 10 s long (as are the RAVDESS files). The clip was a recording of the phrase "I need to take your pulse for", said in a neutral tone. The network misclassified Mr Bush as a female. This may be because he has a distinct accent, unlike those in the training recordings.

3.3 Fusion

Due to time constraints, I have not been able to fuse the two models. However, I have intentionally designed the gender classifier to work with mel spectrograms in the same format as those used for the emotion classifier, to facilitate fusion. As the emotion classification problem is more complex, and the emotion classifier is less successful than the gender classifier, I believe that a suitable fusion paradigm would involve taking the prediction from gender classification, and using this as metadata to be input to emotion classification, potentially improving the emotion classifier's effectiveness.

4 Conclusion

I have implemented an existing CNN for speech emotion classification which adds noise to improve robustness and prevent overfitting, and improved its performance through hyperparameter tuning, particularly raising the learning rate by 10%. This lead to an F1-score increase of 0.027, to 0.856. I have also created my own simple CNN for gender classification, and after tuning this can achieve a test F1-score of 0.943.

References

[1] A. Aljanaki, F. Wiering, and R. C. Veltkamp. "Studying emotion induced by music through a crowdsourcing game". In: *Information Processing & Management* 52.1 (2015), pp. 115–128. DOI: 10.1016/j.ipm.2015.03.004.

[2] F. R. S.R Livingstone. "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English". In: *PLOS ONE* 13.5 (2018). DOI: 10.1371/journal.pone.0196391.

[3] J. Salamon, C. Jacoby, and J. P. Bello. "A Dataset and Taxonomy for Urban Sound Research". In: *22nd ACM International Conference on Multimedia (ACM-MM'14)*. Orlando, FL, Nov. 2014, pp. 1041–1044.

[4] S. Adebiyi. "An Emotion Based Music Recommender System Using Deep Learning". Figure adapted from: *Figure 1: Implementation Pipeline of Emotion Classification and Music Recommendation*. MSc. Dublin, ROI: National College of Ireland, 2020.

[5] S. Adebiyi. 2020. URL: t.ly/KaoW (visited on Apr. 5, 2022).

[6] S. Becker, M. Ackermann, S. Lapuschkin, et al. "Interpreting and Explaining Deep Neural Networks for Classification of Audio Signals". In: *ArXiv preprint.* (July 2018), p. 3. DOI: 10.48550/ARXIV.1807.03418.

[7] E. Allibhai. *Building a Convolutional Neural Network (CNN) in Keras*. URL: t.ly/k6v4 (visited on Apr. 26, 2022).

[8] B. Phan. *10 Minutes to Building a CNN Binary Image Classifier in TensorFlow*. URL: t.ly/7aI3 (visited on Apr. 26, 2022).

[9] A. Sarraf. "Binary Image Classification Through an Optimal Topology for Convolutional Neural Networks". In: *American Academic Scientific Research Journal for Engineering, Technology, and Sciences* 68.1 (May 2020), pp. 182, 185, 187, 188. URL: t.ly/-f54 (visited on Apr. 26, 2022).

[10] G. E. Hinton, N. Srivastava, A. Krizhevsky, et al. "Improving neural networks by preventing co-adaptation of feature detectors". In: *ArXiv preprint.* (July 2012). URL: t.ly/yALy (visited on Apr. 26, 2022).

[11] N. Srivastava, G. Hinton, A. Krizhevsky, et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56 (2014), pp. 1929–1958. URL: t.ly/7alh (visited on Apr. 26, 2022).

[12] S. Doshi. *Various Optimization Algorithms For Training Neural Network*. Jan. 2019. URL: t.ly/r504 (visited on Apr. 26, 2022).

[13] A. Gupta. *A Comprehensive Guide on Deep Learning Optimizers*. Oct. 2021. URL: t.ly/0SIX (visited on Apr. 26, 2022).

[14] Y. LeCun, L. Bottou, G. B. Orr, et al. "Efficient BackProp". In: *Neural Networks: Tricks of the Trade*. Ed. by G. B. Orr and K.-R. Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, 9-50. DOI: 10.1007/3-540-49430-8_2.

[15] N. S. Keskar, J. Nocedal, P. T. P. Tang, et al. “On large-batch training for deep learning: Generalization gap and sharp minima”. In: 2017.

[16] J. Brownlee. *How to Configure the Number of Layers and Nodes in a Neural Network*. July 2018. URL: t.ly/Gar3 (visited on Apr. 27, 2022).

[17] I. Kandel and M. Castelli. “The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset”. In: *ICT Express* 6.4 (2020), pp. 314, 315. DOI: 10.1016/j.icte.2020.04.010.

[18] T. Bots. *The George W. Bush Public Domain Audio Archive*. After 2004. URL: t.ly/OoP1 (visited on Apr. 26, 2022).