FEATURE SELECTION USING ALTERNATING DIRECTION METHOD OF MULTIPLIER FOR LOW-COMPLEXITY ACOUSTIC SCENE CLASSIFICATION

Lorenz P. Schmidt, Beran Kiliç, Nils Peters

Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
International Audio Laboratories, Erlangen, Germany
{lopa.schmidt, beran.kilic, nils.peters}@fau.de

ABSTRACT

Acoustic Scene Classification (ASC) is a common task for many resource-constrained devices, e.g., mobile phones or hearing aids. Limiting the complexity and memory footprint of the classifier is crucial. The number of input features directly relates to these two metrics. In this contribution, we evaluate a feature selection algorithm which we also used in this year's challenge [1].

We propose binary search with hard constraints on the feature set and solve the optimization problem with Alternating Direction Method of Multipliers (ADMM). With minimal impact on accuracy and log loss, results show the model complexity is halved by masking 50% of the Mel input features. Further, we found that training convergence is more stable across random seeds. This also facilitates the hyperparameter search. Finally, the remaining Mel features provide an insight into the properties of the DCASE ASC data set.

Index Terms— ASC, feature selection, hard thresholding

1. INTRODUCTION

An essential part of the training and inference in machine learning is feature extraction. For Acoustic Scene Classification (ASC), feature extraction is often combined with domain knowledge to make features invariant to pitch changes, amplitude differences, and other modifications [2]. For instance, a Mel-scaled filterbank with additional log transformation and normalization is commonly used. In our DCASE 2022 challenge contribution [1], we saw benefits of using automated features selection to meet the low-complexity model constraint [3]. Here, we explain and further examine this approach.

Studying features and performing subset selection [4] is an experimental step that happens before the actual model training. Several benefits make feature selection interesting to a model developers: First, redundant features increase the computational complexity of the model without benefit to the classification accuracy. For example, in convolutional models, the number of MACs (Multiply-Accumulate) operations scale linearly with the number of features [5]. Second, feature selection makes the model more amenable to pruning as the dimensionality of data is reduced. Third, noisy or weakly correlated features can make the training unstable and increase overfitting. Reducing the number of features avoids modelling the noise of the input and therefore decreases variance in the generalization error. This means that a good model can be found with fewer runs of different seeds. Finally, from a research perspective, selecting features allow us to draw conclusions on the generation process and to interpret patterns of the feature sets. This may give a clue about deficiencies and better feature parametrization.

Sparse estimation methods and the concept of parsimonious representation [6] is an established field for linear models and found recent application for DNNs [7, 8]. Unstructured pruning and feature selection is expressed in linear models as zeroing out model parameters, which also removes correlation between feature and target. In (non-linear) DNNs, this relation is more complex because parameters' magnitudes are not linearly connected to outcomes. Further solutions may differ for each run, making comparisons and drawing conclusions on feature importance difficult. Applying a proximal operator to the iterate directly gives magnitude pruning [9], often seen in DNN methods such as deep compression [10] or lottery ticket [11]. But an abundance of methods exist and are used for compression, including reweighted l_2 penalized techniques [6], log barriers [12], active sets [13] and proximal methods [9].

In Section 2, we state the problem of optimizing with feature cardinality constraint C and derive an algorithm with the Alternating Direction Method of Multipliers (ADMM) [14] and proximal operators [9]. We also explain how binary search is used for finding a bound on C adhering to a threshold on performance loss. We conduct experiments in Section 3, discuss results in Sec. 4, and conclude this study with an outlook in Sec. 5.

1.1. Study findings

- Masking half of the features results in minimal loss of accuracy; discarding them saves half of the DNN model complexity
- Feature selection reduces training variance and improves convergence speed; enabling efficient single-run hyperparameter searches
- Different model architectures vary in selected feature sets
- Mel features above 19 kHz seem to provide unexpected benefits to ASC, with fewer information contained between 9 kHz and 19 kHz

2. FEATURE SELECTION PROBLEM

In this section, we will derive a feature selection method using ADMM. Instead of using an explicit penalty (e.g. l_1 norm on the feature support), we chose to have hard constraint on the feature cardinality. We do this to make the application applicable to binary search as we are interested in the feature sub-set and the amount of features we can mask. The proximal operator makes our method applicable to a number of different constraints, including matrix constraints [15] or mixed ℓ_1/ℓ_2 with elastic net regularization [9].

The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg and Fraunhofer IIS.

2.1. Feature selection using ADMM

We split the DNN loss function and non-smooth constraint with dual decomposition and solve the sub-problem of constraining the feature support with an proximal operator.

Our objective is to minimize the loss function of a model while fulfilling constraints on the support of the feature set. The optimization problems reads then as

$$\min_{\theta,\mathbf{w}} f_\theta(\mathbf{w} \circ \mathbf{X}) \;\text{ s.t. }\; \text{card}(\mathbf{w}) \leq C, \tag{1}$$

where $\mathbf{X}$ is the input spectrogram, $\mathbf{w}$ the feature support vector (masking by rows with $\circ$), θ the model weights, card$(\mathbf{w})$ the number of non-zero elements, and C is the hard constraint on the feature set. We move the hard constraint into the loss function with indicator

$$I_C(\mathbf{w}) = \begin{cases} 0 & \text{if } \text{card}(\mathbf{w}) \leq C \\ +\infty & \text{otherwise} \end{cases} \tag{2}$$

and split along the feature mask $\mathbf{w}$ by introducing equality constraints

$$\min_{\theta,\mathbf{w}} f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) \quad \text{s.t.} \quad \mathbf{w} = \hat{\mathbf{w}}. \tag{3}$$

The optimization problem contains the smooth and differentiable term of the DNN loss function and the non-smooth indicator function $I_C(\hat{\mathbf{w}})$. While this cannot be solved directly via gradient descent, we can relax the problem by introducing a Lagrangian multiplier for the equality constraint $\mathbf{w} = \hat{\mathbf{w}}$ and optimizing those via gradient ascent. We can derive the augmented Lagrangian as

$$\mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}, \mu) = f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) + \langle \mu, \mathbf{w} - \hat{\mathbf{w}} \rangle + \frac{\rho}{2} \|\mathbf{w} - \hat{\mathbf{w}}\|_2^2 \tag{4}$$

with μ the dual variable and ρ smoothing factor of a Moreau envelope [16]. A stable saddle point does not have to exist for non-linear DNNs. In practise, treating ρ as a hyperparameter, we observe convergence. The scaled version with dual variables $\mathbf{u} = \frac{1}{\rho}\mu$ is then

$$\mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}, \mu) = f_\theta(\mathbf{w} \circ \mathbf{X}) + I_C(\hat{\mathbf{w}}) + \frac{\rho}{2} (\|\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u}\|_2^2 - \|\mathbf{u}\|_2^2) \tag{5}$$

and we can optimize over $(\theta, \mathbf{w})$, $\hat{\mathbf{w}}$ and μ. In each cycle $k = 0, 1, \ldots$ we update the variables separately and repeat until convergence. We use the iteration number k for variables, which are fixed in the following section. For $(\theta, \mathbf{w})$ we can see that

$$\nabla \mathcal{L}_p(\theta, \mathbf{w}, \hat{\mathbf{w}}^k, \mu^k) = \nabla f_\theta(\mathbf{w} \circ \mathbf{X}) + (0, \rho(\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u})) \tag{6}$$

and therefore the gradient of θ is unchanged, but $\mathbf{w}$ is influenced by the regularization term. In practice the PyTorch loss function [17] is extended with the regularization term, while derivatives are generated with automatic differentiation.

The sub-problem of $\hat{\mathbf{w}}$ has a closed-form solution. The closest projection onto the support constraint

$$\min_{\hat{\mathbf{w}}} I_C(\hat{\mathbf{w}}) + \frac{\rho}{2} \|\mathbf{w} - \hat{\mathbf{w}} + \mathbf{u}\|_2^2, \tag{7}$$

gives a closed-form solution $\hat{\mathbf{w}} = \Pi_C(\mathbf{w} + \mathbf{u})$ [9]. Basically, we keep the C largest elements of $\mathbf{w} + \mathbf{u}$, while setting the rest to zero. Finally, optimizing for $\mathbf{u}$ results in gradient ascent of the dual variable and in total we have

$$\begin{aligned} (\theta^{k+1}, \mathbf{w}^{k+1}) &= (\theta^k, \mathbf{w}^k) - \gamma_k \nabla f_\theta(\mathbf{w} \circ \mathbf{X}) - (0, \rho(\mathbf{w} - \hat{\mathbf{w}}^k + \mathbf{u}^k)) \\ \hat{\mathbf{w}}^{k+1} &= \Pi_C(\mathbf{w}^{k+1} + \mathbf{u}^k) \\ \mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{w}^{k+1} - \hat{\mathbf{w}}^{k+1}, \end{aligned} \tag{8}$$

with γ_k the learning rate of gradient descent. Observe that we keep the penalty ρ fixed, which in turn means that we have a fixed learning rate for the duals.

2.2. Learning phase

In Section 2.1, we use ADMM to derive an optimization routine for our hard constraint. Because we make no assumption about the model and we cannot ensure that a saddle point even exists for our Lagrangian, we have to take care of the optimization process. First, we pre-train the model with all features and use those weights as initialization for our feature search. Second, we then apply Eq. 8 for a number of epochs. The learning rate of gradient ascent is fixed, while that of gradient descent anneals to zero over the learning cycle. This ensures that the model loss dominates at the beginning, reducing to zero those features unnecessary for good performance. At the end of the training cycle, the penalty of the ADMM algorithm dominates that of the model, forcing the constraint on feature support to be fulfilled, even though the performance may suffer. Finally, after finding the optimal feature mask, we prune unused features and re-train for a small number of epochs (less than 10) to fine-tune the model.

3. EXPERIMENTS

Our experiments use the DCASE 2022 Task 1 [3] split with samples for training (139619), validation/development (29680), and testing (29680). Each sample belongs to one of 10 different acoustic scenes, captured at 44.1 kHz in one of 12 European cities by 3 real (A, B, C) or 6 simulated recording devices (S1-S6).

3.1. BC-ResNet model

As a state-of-the-art ASC model, we use a BC-ResNet [18] architecture, specifically the parametrization of BC-ResNet Mod-8 [19] which performed excellent in the DCASE 2021 challenge.

3.1.1. Pre-training

Mel-scaled spectrograms with 512 bands are extracted from full-band audio input. We use a frame length of 93 ms (4096 samples), an overlap of 30 ms (1323 samples) and Hamming weighting. We apply a logarithmic transformation and residual normalization [19] with $\lambda = 0.1$ to generalize across different devices.

The input features are augmented to avoid overfitting. We use a random roll of 40% of the signal length. We also use Specaugment [20] in frequency domain with mask parameter of 20 and Mixup [21] with $\alpha = 0.2$. We apply stochastic gradient descent (SGD) to the model and train for 60 epochs. The learning rate increases to 0.01 in a warmup phase (3 epochs) and then decreased to 0.0001 in the remaining 57 epochs. We use momentum of 0.9, weight decay of 0.001, and a mini-batch size of 64.

We train the BC-ResNet with eight different random seeds. The accuracy during training is depicted in Fig. 1 and the final performance can be seen in Table 1. With 512 Mel bands and 1 sec audio per sample the model has a computational complexity of 300 k parameters and 102 MMACs. We select the best performing model (in terms of the log loss on the testing data) as the initial model weights to use in the following experiments.

3.1.2. Optimizing with feature constraints

After pre-training a BC-ResNet model, we are applying Eq. 8 with a given feature constrain C. The primal $\mathbf{w}^0$ is initialized with ones and projected; and dual variables $\hat{\mathbf{w}}^0$, $\mathbf{u}^0$ with zeros. This penalizes all features in the first step; the second step proceeds normally by masking smallest magnitudes. We update the model weights with an initial learning rate of 4×10^{-3} and cosine annealing for five epochs. After finding a set of features and masking them, we fine-tune the model for another five epochs, resetting the optimizer to learning rate 4×10^{-3} first.

The convergence of ADMM highly depends on choice of ρ. We treat ρ as a hyperparameter and found $\rho = 3 \times 10^{-4}$ to be a good trade-off between fast convergence and stability. Unfortunately, the convergence of the variables depends not only on ρ but also on the number of masked features. As illustrated in Fig. 1, we observe that when masking more features, the initial convergence seems to be faster. When masking more features than realizable (without losing performance), the convergence continues only once the model's learning rate is annealed enough. As we use cosine annealing, this happens in the later phase of training, where the regularization term introduced by ADMM dominates the update step of θ and $\mathbf{w}$. This ensures that the cardinality constraint in Eq. 1 is fulfilled, though we may lose performance. We finally see no difference in convergence of primal and dual variables: both converge in similar rates and to similar residuals (see Fig. 2).

3.1.3. Finding a good constraint on the feature number

Choosing an appropriate C prior running feature selection is difficult. If too high, then the model remains too complex and training speed suffers; if too low, then additional error is introduced

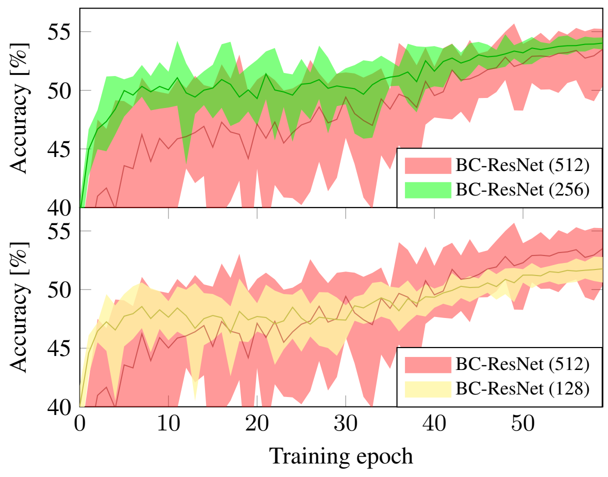

Figure 1: Variance and error during training for validation accuracy. Max, min, and geometric means are depicted for eight runs with different random seeds are shown. Reducing number of features from 512 to 256 improves stability and reduces variance, selecting only 128 features affects performance.

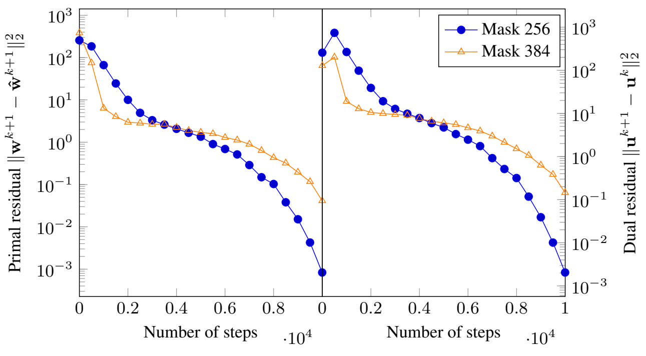

Figure 2: Convergence of primal (left) and dual (right) variables for masking 256 and 384 features. Masking more features initially increases the influence of the regularization term, but requires more iterations to reach low residuals.

(see also Sec. 4.1). Therefore we apply the feature selection in context of a meta heuristic by introducing a maximal degradation $\epsilon = 3 \times 10^{-2}$ of log loss for our pre-trained model. The heuristic performs binary search, halving upwards if the log loss is under our threshold ϵ; downwards, if not. For reliable results, we run the experiment with 8 different random seeds and select the best performance. Once the step size is below 8, we step alternate to stay below the threshold and explore different log loss in that region. Alternatively, (in a more conservative search) the average over all runs could be compared to the absolute threshold. In the end, we conclude with $C = 274$ masked features (see also Fig. 3).

3.2. Linear model

For comparison with the non-linear BC-ResNet model, we also train a linear Support Vector Classifier (SVC) [22]. The SVC performs inference with temporally global features. We reduce the time axis of our Mel spectrogram with moments up to fifth order (mean, variance, skewness, kurtosis, and hyperskewness for each band), resulting in 2560 features per data sample.

Similar to the previous experiment, we apply l_1 regularization during feature selection. The penalty is treated as a hyperparameter. A search found that the model gives the best accuracy on the testing data for $\nu = 100$. We have seen almost no variance for the final

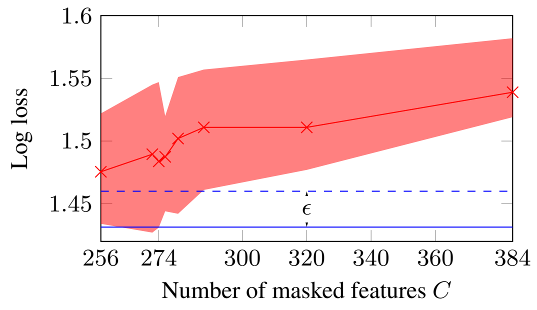

Figure 3: Log loss on validation set during binary search. Max, min, and geometric means of eight runs with different random seeds are shown. The dashed line indicates the threshold $\epsilon = 0.03$ of the log loss to the performance of the pre-trained model (see Sec. 3.1.1).

Model	MMACs	Acc. (%)	Log Loss
Baseline [3]	29.23	42.90	1.575
SVC ($\nu = 100$)	0.026	43.79	1.780
BC-ResNet (512)	101.89	54.24	1.370
BC-ResNet (256)	51.53	54.10	1.496
BC-ResNet (128)	26.35	52.80	1.613

Table 1: MACs and performance on development set for best models. The number of features is indicated in brackets behind the model.

testing results, making the need to do multiple runs unnecessary. Compared to the DCASE 2022 baseline [3], this SVC has a better model accuracy and a worse log loss, but requires significantly fewer MACs (see Table 1).

4. DISCUSSION

4.1. Training variance and model reduction

Having many noisy or weakly correlated features to the targets increases the risk of overfitting. We see this trend in the variance of our validation results. As can be seen in Fig. 1 for the BC-ResNet without feature selection (512 features), the model uses noisy features. Even at the end of training, its accuracy differs by several percentages. Consequently, finding a good solution requires multiple training runs, which is time- and energy-consuming. Halving the number of features to 256 has several effects. First, in the initial training phase, the model converges much faster on average as the variance is reduced drastically. Then the accuracy approximately stays the same until the final phase. All runs with different seeds give a good final result. Thus, hyperparameter searches can be executed much more efficiently. This reduces experimentation time and improves sustainability of our model search.

When reducing the number of Mel features to 128, variance slightly decreases between epochs 30 to 50, but the final model performance was compromised. We decrease variance, but underfit the dataset by using too few features. This behavior is also supported by our binary search, which suggested 278 masked features, i.e., 234 remaining features (see Fig. 3).

The BC-ResNet model uses only convolutional layers [18]. Consequently, the total number of MACs scales linearly with the number of features. By selecting only 50% of the features, we also halve the number of operations from $101.89\,\mathrm{MMACs}$ to $51.53\,\mathrm{MMACs}$ without significant change in performance. When further reducing the feature count, the log loss increases significantly (see Table 1).

4.2. Interpretation of feature selection

Selection allows making interpretation about the feature set. This is different to feature extraction, where all features are retained and only their representation is changed. Fig. 4 depicts the histogram of the selected feature in our experiments. For the BC-ResNet models, most of the information is contained below $8\,\mathrm{kHz}$. This matches with the literature that $16\,\mathrm{kHz}$ sampling rate (leading to a $8\,\mathrm{kHz}$ lowpass filtering) gives good ASC results. However, we also tested pre-training of the BC-ResNet with a naïve selection of all features below $8\,\mathrm{kHz}$, but model performance was below that of BC-ResNet (128).

The feature importance around $\approx 19.8\,\mathrm{kHz}$ for all BC-ResNet models is unexpected. It may be related to some recording artifacts by some device types. Masking those bands may improve generalization. Other features are more plausible, e.g., we find that features below $500\,\mathrm{Hz}$ are always selected (as they relate to low frequency or harmonics) and the region between $2.5\,\mathrm{kHz}$ and $8\,\mathrm{kHz}$ contain a lot of information as well, e.g., for consonants in speech.

Also visible in Fig. 4, one can see that feature selection subsamples the frequency bins, suggesting that the band width of $\approx 5.4\,\mathrm{Hz}$ is too small. However, features of very low frequencies (below $60\,\mathrm{Hz}$) are fond to be also relevant. With increasing masking, the feature selection in regions above $500\,\mathrm{Hz}$ starts to subsample even more. This is most prominent when examining the 128 remaining features. In this case, only a handful of features are selected.

When comparing the BC-ResNet results to the linear SVC model (upper row of Fig. 4), we see that the regions below $500\,\mathrm{Hz}$ and between $1.5\,\mathrm{kHz}$ to $8\,\mathrm{kHz}$ contain most of the important information. On the other hand, the linear model uses higher frequencies above $9\,\mathrm{kHz}$, but not the prominent peak at $19.8\,\mathrm{kHz}$ selected by BC-ResNet. This may suggest that the moments of the SVC may model different information than the convolutional layers of BC-ResNet. This indicates that feature selection is partially model-dependent. More work is needed to better understand this relationship.

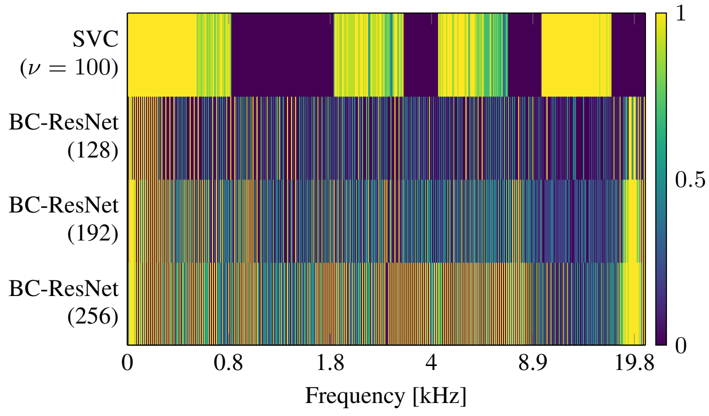

Figure 4: Selected frequency bands for SVC (upper row) and BC-ResNet models across input feature sizes. The selection patterns are averaged across all 8 runs. The color indicates the number of occurrences a feature was selected; with yellow color representing features that are used in all runs. For the SVC model a frequency band counts when at least one moment is selected.

5. CONCLUSION

We showed that feature selection is an important experimental step when developing efficient machine learning models. In addition to providing insights into the working of signal processing, it also improves stability and makes hyperparameter search more efficient. We applied this method in the DCASE 2022 challenge to reduce the total number of MACs [1]. In future, we aim to further examine how features map to different scenes, devices, and cities. This would allow for optimizing methods and models.

6. REFERENCES

[1] L. Schmidt, B. Kiliç, and N. Peters, "Structured filter pruning and feature selection for low complexity acoustic scene classification," DCASE2022 Challenge, Tech. Rep., June 2022.

[2] J. Andén and S. Mallat, "Deep scattering spectrum," *IEEE Transactions on Signal Processing*, vol. 62, 04 2013.

[3] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, "Low-complexity acoustic scene classification in DCASE 2022 challenge," 2022. [Online]. Available: https://arxiv.org/abs/2206.03835

[4] R. Kohavi and G. H. John, "Wrappers for feature subset selection," *Artificial Intelligence*, vol. 97, no. 1, pp. 273–324, 1997, relevance. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S000437029700043X

[5] M. Taghavi and M. Shoaran, "Hardware complexity analysis of deep neural networks and decision tree ensembles for real-time neural data classification," in *2019 9th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2019, pp. 407–410.

[6] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Found. Trends Mach. Learn.*, vol. 4, no. 1, p. 1–106, jan 2012. [Online]. Available: https://doi.org/10.1561/2200000015

[7] Y. Li, C.-Y. Chen, and W. Wasserman, "Deep feature selection: Theory and application to identify enhancers and promoters," in *2015 Annual International Conference on Research in Computational Molecular Biology (RECOMB)*, 04 2015.

[8] M. Wojtas and K. Chen, "Feature importance ranking for deep learning," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 5105–5114. [Online]. Available: https://proceedings.neurips.cc/paper/2020/file/36ac8e558ac7690b6f44e2cb5ef93322-Paper.pdf

[9] N. Parikh, S. Boyd, *et al.*, "Proximal algorithms," *Foundations and trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[10] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," 2015. [Online]. Available: https://arxiv.org/abs/1510.00149

[11] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," 2018. [Online]. Available: https://arxiv.org/abs/1803.03635

[12] H. Kervadec, J. Dolz, J. Yuan, C. Desrosiers, E. Granger, and I. B. Ayed, "Constrained deep networks: Lagrangian optimization via log-barrier extensions," 2019. [Online]. Available: https://arxiv.org/abs/1904.04205

[13] J. Kim and H. Park, "Fast active-set-type algorithms for l1-regularized linear regression," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Y. W. Teh and M. Titterington, Eds., vol. 9. Chia Laguna Resort, Sardinia, Italy: PMLR, 13–15 May 2010, pp. 397–404. [Online]. Available: https://proceedings.mlr.press/v9/kim10a.html

[14] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[15] J. E. Cohen, "Computing the proximal operator of the ℓ_1 induced matrix norm," 2020. [Online]. Available: https://arxiv.org/abs/2005.06804

[16] C. Planiden and X. Wang, *Proximal Mappings and Moreau Envelopes of Single-Variable Convex Piecewise Cubic Functions and Multivariable Gauge Functions*. Cham: Springer International Publishing, 2019, pp. 89–130. [Online]. Available: https://doi.org/10.1007/978-3-030-11370-4_5

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[18] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," 2021. [Online]. Available: https://arxiv.org/abs/2106.04140

[19] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI submission to DCASE 2021: Residual normalization for device-imbalanced acoustic scene classification with efficient design," DCASE2021 Challenge, Tech. Rep., June 2021.

[20] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," in *Interspeech 2019*. ISCA, sep 2019. [Online]. Available: https://doi.org/10.21437%2Finterspeech.2019-2680

[21] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2017. [Online]. Available: https://arxiv.org/abs/1710.09412

[22] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.