Using Representation Learning and Out-of-domain Data for a Paralinguistic Speech Task

Benjamin Milde and Chris Biemann

Language Technology Group, TU Darmstadt, Germany
milde,biem@lt.informatik.tu-darmstadt.de

Abstract

In this work, we study the paralinguistic speech task of eating condition classification and present our submitted classification system for the INTERSPEECH 2015 Computational Paralinguistics challenge. We build upon a deep learning language identification system, which we repurpose for general audio sequence classification. The main idea is that we train local convolutional neural network classifiers that automatically learn representations on smaller windows of the full sequence's spectrum and to aggregate multiple local classifications towards a full sequence classification. A particular challenge of the task is training data scarcity and the resulting overfitting of neural network methods, which we tackle with dropout, synthetic data augmentation and transfer learning with out-of-domain data from a language identification task. Our final submitted system achieved an UAR score of 75.9% for 7-way eating condition classification, which is a relative improvement of 15% over the baseline.

Index Terms: speech classification, computational paralinguistics, neural networks, deep learning, transfer learning, data augmentation

1. Introduction

The study of paralinguistic speech involves perceptual speech phenomenons that lie beyond the pure transcriptional content of spoken speech [1]. The term paralanguage [2], literally 'alongside language', was coined by linguist Archibald Hill. With the advent of computational paralinguistics, such phenomena are attained by automatic means. Automated gender discrimination, age estimation [3] and emotional state classification [4] are some examples with a range of diverse applications in e.g. health care, commerce (advertising) and education. Computational paralinguistic speech tasks can encounter data sparsity problems, as it can be more challenging to find or generate larger amounts of labelled training examples for the desired domain than in other areas of speech research.

It is also not easy to argue what constitutes good features for a specific paralinguistic regression or classification task and for some of these tasks, even human performance can be suboptimal [5]. However, a specific set of speech features developed in a particular paralinguistic domain can often be applied to another paralinguistic domain with success [6]. It would certainly be desirable, if features could adapt automatically to a new problem domain, instead of the need to manually engineer and tune features. The recent resurge of interest in neural network approaches to image, audio and text classification problems enables such a new learning paradigm: promising feature representations can also be learned automatically from the data of the problem domain [7].

In this paper, we describe the methods we used for our entry in the Eating Condition (EC) challenge of the Interspeech 2015 Computational Paralinguistics ChallengE (ComParE2015) [8]. We use the iHEARu-EAT corpus [9], which consists of 945 training utterances and 469 test utterances. The task is to predict the kind of food someone eats while s/he speaks and it is posed as a 7-class sequence classification problem with six specific kinds of food and one class for speech without food (No Food, Banana, Crisp, Nectarine, Haribo, Apple, Biscuit).

Nonetheless, we think our ideas should be transferable to other paralinguistic and audio classification tasks. We particularly address the challenge of having only a few labelled example sequences for each class and show how the training process makes meaningful use of out-of-domain data with transfer learning [10, 11] and data augmentation with synthetic training data examples. Our main idea is to train local classifiers that automatically learn representations on smaller windows of an audio sequence and to combine multiple classifications from such windows to obtain a classification result for a full sequence input. Alongside better classification performance compared to systems trained only on per-sequence openSMILE [12, 13] baseline features, our window approach has the advantage that it also enables us to extract exact positions of an audio sequence that are most relevant for its assigned class.

2. Methods and approaches

Our approach bears similarity to the proposed deep learning architecture for language identification in [14]. We have obtained good results by replacing Deep Neural Networks (DNNs) in this pipeline with Convolutional Neural Networks (CNNs) [15, 16, 17], which have demonstrated recognition accuracy better than or comparable to humans in several visual recognition tasks [18, 19].

2.1. CNN-based audio sequence classification

Similarly to the recent work in applying CNNs to the audio and speech domain [21, 22], we interpret overlapping windows in a frequency spectrum representation as fixed-dimensional 2D input to the CNNs (comparable to grey scale images). We have used windows of 40-dimensional log mel filterbank features (FBANK) as input to the classifier. End-to-end representation learning on raw audio would also be possible, but recent results suggest that providing some kind of spectrogram input gives better performance [23]. Each FBANK vector is calculated from a time frame of 25 ms of raw audio and we use a window of +/- 5 feature vectors around a central feature vector as input to the CNN. A window has thus a height of 40 and a width of 11 elements, totalling 440 inputs. We shift the FBANK window by two frames through all FBANK frames of the signal.

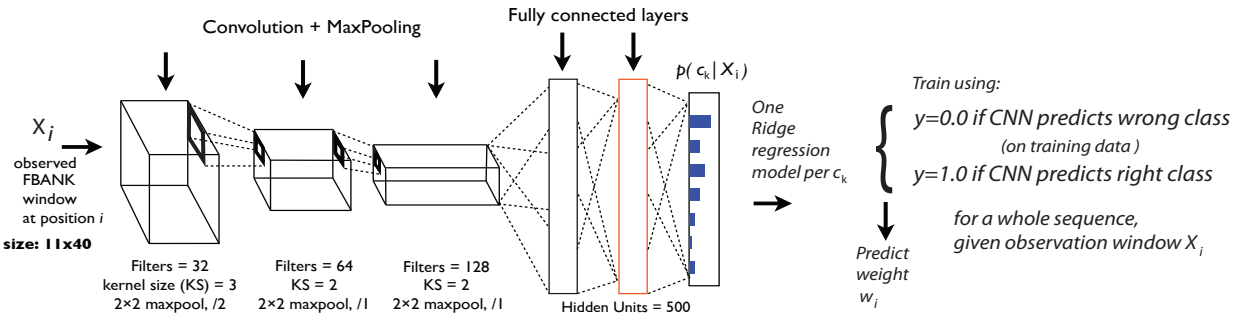

Figure 1: The CNN architecture used for learning on local FBANK patches X_i of a full sequence X (drawn in the style of [20]). Contribution weights w_i are learned separately with linear models, one for each class, which assign a weight to the local classification's contribution to the full classification of a sequence.

Figure 1 illustrates our CNN architecture, along with a weight prediction scheme using linear models. We follow the standard CNN design of having alternating convolutional and maxpool layers followed by fully connected layers [17]. Initially, we used three convolutional and maxpool layers followed by two fully connected layers. This is much smaller than proposed architectures for the ImageNet classification task [17, 19] to account for the smaller input size of the CNN and the smaller amount of training data available in iHEARu-EAT. We also use smaller convolution kernel sizes of 3x3 and 2x2, to achieve an implicit regularisation effect [24]. One particular challenge of audio sequence classification is that the length of unseen sequences is not fixed. We have implemented a simple majority voting strategy that selects the highest count of individual class predictions across all windows of the sequence and a weighted majority voting that learns contribution weights with linear regression models that we describe in the next subsection.

2.2. Weighted majority voting

We use one Ridge linear regression [25] model per class c_k, to predict the weight of a window's prediction towards the full prediction of the sequence. A CNN model is trained on all windows X_i of all audio sequences $seqs = seq_1, seq_2, ..., seq_n$, $X_i \in seq_j$. Each window is assigned the output class c of the sequence j it belongs to. Then, for each X_i we let the model predict the most probable class c_k given its window of observation. We assign $y = 0.0$ for each wrong prediction and $y = 1.0$ for each successful prediction on the training data the model was trained on. The assumption here is that some windows of a sequence contain no discriminative information (e.g. only silence). A linear regression model m_{c_k} should then learn the connection between all predicted class probabilities of its class c_k given the window of observation X_i and weights $w_i = y$. Ideally, these weights should be closer to 0.0 if the predicted input class probabilities of the observation X_i are likely to contribute to an incorrect classification and close to 1.0 if it supports the full sequence's true class. Then, we add the predicted weight values w_i towards a final vote of class c_k for seq_j.

2.3. Dropout

We employ dropout [26] as a first measure to counter overfitting in training our networks. It has been specifically proposed for cases where labelled data is scarce and thus where overfitting is likely. It works by randomly omitting a certain percentage of nodes in the network at training time, while using the full network at test time.

2.4. Data Augmentation

Data augmentation has been widely used in neural network based pattern recognition tasks [16, 27, 17]. In image processing, data augmentation techniques are usually intuitive and transformations such as translation, deformation and reflection [17] have led to significant improvements in recognition accuracy. Inspired by these image augmentation techniques, we employ a simple audio augmentation technique: We artificially generate pitch shifted versions of the original training files and add them to the training set. We make use of the Rubberband utility[1] to generate new files shifted by one semitone upwards and downwards, without changing the length of the recordings. We set the "crispness" level to 6, intended for drum tracks, as this produced the best sounding pitch-shifted speech output and was deemed somewhat fitting for smacking noises.

2.5. Transfer learning

The idea of training a neural network on a related problem first and transferring its weight configuration for a target problem instead of random initialization is an old one [28, 29]. However, literal and naive weight transfer has been shown to not only be ineffective but also detrimental to the classifiers performance [29]. These results have now been challenged on several visual recognition tasks with deep neural networks, where representation transfer has been shown to be astoundingly effective [30]. Azizpour et al. [31] have recently systematically identified the list of factors that affect the transferability of CNN representations and of weight configurations for visual recognition tasks, with exhaustive experimental evidence showing how these factors should be set.

An important factor is, unsurprisingly, how related the two problems are. It is then very important, dependent on the relatednesses of the tasks, how many of the first layers are transferred, while the remaining layers are randomly initialized as usual. We have chosen the Voxforge[2] corpus for transfer learning as it is freely available and easily accessible. We build

[1] http://breakfastquay.com/rubberband/
[2] http://www.voxforge.org/

a 7-class language classification problem, where the sequence should be classified as German, French, Spanish, Italian, Portuguese, Dutch or Russian. We selected 3000 utterances for each language from the Voxforge data set, which is roughly 30 times larger than the iHEARu-EAT dataset. We then pretrained our CNN model on this problem first and then transferred the weight matrices of the model to a new CNN model, which we train on the iHEARu-EAT data. We have both tested the naive transfer learning approach where we transfer all layers and one where we only transfer the weights of convolutional layers, but initialize the fully connected layers randomly.

2.6. Hyperparameters

For training DNNs and CNNs we use Nesterov's Accelerated Gradient Descent (NAG, Nesterov momentum) [32, 33]. For all random weight initializations, we have chosen He-initialization, as described in [19]. We use categorical cross-entropy as the objective loss function, i.e. the measure of error between computed and desired target outputs of the training data, which we minimize. One of the downsides of gradient based optimization methods, which are primarily used in DNNs and CNNs, is that they usually introduce additional hyperparameters that govern the behaviour of the optimization techniques. We start with reasonable defaults and follow best practices: 0.02 is chosen as the start learning rate and 0.9 as the start momentum. For transfer learning, we slightly reduce the initial learning rate to 0.01 [31]. Then, we linearly reduce it with each epoch, so that the learning rate for the last epoch is 0.0001. Similarly, momentum is linearly increased [33], so that it is 0.999 in the last epoch. We track performance of the network for each epoch on 1% of all patches, which can be seen as held-out classification performance on known speakers. We train all our cross validation models for 400 epochs. We set dropout to 0.1, 0.2 and 0.3 for the first, second and third convolutional layers and 0.5 for all fully connected layers.

3. Results

We implemented the proposed approaches and methods in the previous section using Nolearn+Lasagne[3] and Scikit-learn [34] in Python. Since Lasagne is based on the Theano [35] architecture, we could use GPUs to speed up computations. The official development evaluation is unweighted average recall (UAR) with leave-one-speaker-out cross-validation (LOSO-CV). This proved to be computationally impractical for us, as computing one CNN model can take up to one day on a reasonable GPU (Nvidia Geforce 970), depending on the number of training epochs used. We opted to do 5-fold cross validation over the speakers instead, with 400 epochs of training. Table 1 shows our results for this cross validation method. While this allowed us to test different ideas quicker, it makes our results not comparable to the baseline LOSO-CV evaluation of the challenge (61.3% 7-class UAR) [8]. We thus recomputed the baseline results on exactly the same split with 3 different classifiers: SVM (C=0.001), Random Forests (RF) and a 3-layer DNN with dropout. CNN results were computed with the architecture shown in Figure 1. We performed mean normalization prior to training on the FBANK patches. For all results shown, we have kept the number of hidden units in the fully connected layers at 500. Preliminary experiments showed that setting hidden units to 1000 or 250 in the two last fully-connected layers of the CNN yielded worse results.

[3] `https://github.com/dnouri/nolearn` and `https://github.com/benanne/Lasagne`

Method	UAR 7-class	UAR 2-class
Baseline features (SVM)	56.0% (+/- 2.7%)	**94.3%** (+/- 2.3%)
Baseline features (RF)	53.6% (+/- 2.9%)	91.4% (+/- 4.3%)
net0: Baseline features (DNN)	**59.2%** (+/- 4.2%)	93.8% (+/- 3.6%)
net1: CNNs (simple maj. vote)	57.8% (+/- 3.6%)	94.4% (+/- 1.8%)
net2: net1+dropout	61.6% (+/- 6.1%)	94.1% (+/- 2.8%)
net3: net2+weighted maj.	63.1% (+/- 1.4%)	95.0% (+/- 2.2%)
net4: net3+naive transfer learning	62.2% (+/- 3.6%)	95.4% (+/- 2.2%)
net5: net3+transfer learning	**65.5%** (+/- 2.2%)	**96.6%** (+/- 1.4%)
CNN ensemble (net0+net2+net3+net5)	**67.2%** (+/- 2.6%)	**96.3%** (+/- 2.0%)

Table 1: UAR mean and standard deviation with 5-fold speaker independent cross validation, using the training data of iHEARu-EAT. In each split the data of 16 speakers were used for training the classifier(s) and the utterances of the remaining 4 speakers were used to evaluate classification performance.

Without any regularization method, a CNN on FBANK features does actually perform worse than a regularized DNN on per-sequence baseline features (net1). However, if we use dropout, we are able to out-perform the results obtained with the baseline features (net2). Replacing the simple majority voting with our weighted majority voting based on linear regression models also improves performance (net3), but using transfer learning naively by transferring all weights from the source task to the target task produces worse results (net4). However, when the last two fully connected layers are not transferred, but initialized randomly, we found transfer learning to be surprisingly effective (net5), showing a large single-technique contribution. If we combine the class probability estimates of differently trained CNN models by their harmonic mean, we can further improve our score, beyond the performance of any single method (ensemble).

The first ensemble model for predicting on the test set was retrained on the full training set with the same parameters. With 400 training epochs, this produced a 70.0% 7-class UAR on

Submitted system	UAR	Acc.
8-layer CNN ensemble (400 epochs)	70.0%	70.8%
+ 10-layer CNN ensemble (3000 epochs)	73.6%	74.4%
+ 10-layer CNN with data aug. (3000 epochs)	75.4%	76.1%
+ Log. regression on CNN features (trained with SGD, 1000 epochs)	**75.9%**	**76.5%**

Table 2: 7-class UAR and accuracy scores on the official test set, as reported by the official challenge test submission system. Each new submission is an ensemble which also includes previous models. Five individual submissions were allowed.

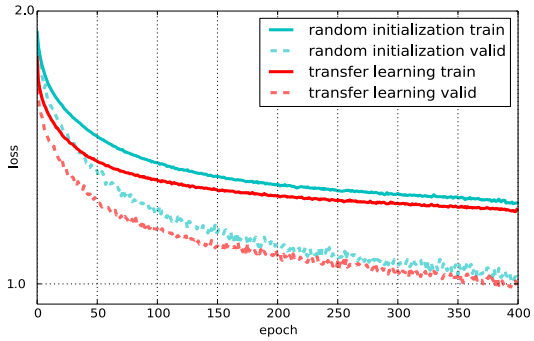

Figure 2: Minimizing the objective function: training loss per epoch for a CNN with completely random initialization and one with transferred weights. Validation loss is measured on held-out data from already known speakers.

the official test set, see Table 2. We proceeded to make some changes which extended training time considerably ($>$ 5 days per model) and were thus only evaluated on the test set: We added one additional convolutional layer and pooling layer before the network's fully connected layers with the same parameters as its previous convolutional and pooling layer and extended the input window to 18 frames with 60 dimensional FBANK features. We also trained the network for 3000 epochs instead of 400 epochs. An ensemble network trained with this larger network raises our test score to 73.6% UAR. Finally, adding a CNN to the ensemble that was additionally trained on augmented pitch-shifted training examples and logistic regression classifiers trained on CNN feature embeddings resulted in an UAR of **75.9%**, which is 15% better than the baseline UAR of 65.9% reported in [8]. Figure 2 compares validation loss and training loss per epoch for CNNs with completely random initialization and transferred convolutional kernel weights from the Voxforge language identification task, where the latter gives an advantage in all epochs.

3.1. Insights

Figure 3 shows an example classification on an at training time unseen speaker. The correct answer is 'Crisp', which the model got right. Note that the 'No food' class is relatively inactive for almost the entire utterance, which suggests that a small window of a whole utterance is sufficient to decide if someone eats while he speaks or not. The class 'Crisp' has several distinct spots where classification probabilities are very high, which correspond to characteristic crunch noises in the audio. Our classification pipeline is capable of producing insights, as it can point to sound examples and exact positions in an utterance which are most characteristic and discriminative of its class.

4. Conclusion

The eating challenge (EC) provides an interesting audio classification task, where relatively few samples per class are available, but good automated classification performance can still be attained. Prior to 2012, when neural networks were renowned for their propensity to overfit, they would have probably not been considered for such a learning task. Without techniques like dropout and transfer learning, it would have been difficult to beat the baseline results. On the cross validation results, the use of additional speech data via transfer learning to initialize early

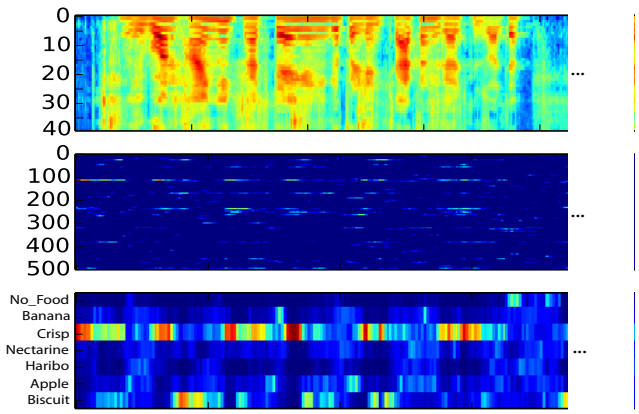

Figure 3: Classification of (yet unseen) train utterance `train_0112.wav` by a CNN model. The correct class is 'Crisp'. From top to bottom: FBANK features, CNN embedding and the 7 class probabilities for its corresponding window. Orange/red colours represent higher values, while dark blue represents zero.

layers of the CNN showed a large single-technique improvement and overall best performance was achieved with a voting scheme between several CNNs. At the expense of a longer training time, we could also further improve our test set result with data augmentation and a bigger network, yielding a final improvement of 15% relative to the baseline score.

To the best of our knowledge, this is the first time that a convolutional neural network has been applied to a paralinguistic speech task. We have opted to a pipeline where individual classification results are averaged across a sequence, to produce a combined classification result for the full sequence. Examination of the feature representation of the last fully connected layer suggests that such an automatically learned representation is fairly sparse. Our results concerning the transferability of convolutional neural network weights are in line with newer results and insights obtained on visual object recognition tasks [31]: It is paramount to only transfer the first layers to the target task and to initialise the last layers of the target network at random, since these are very specific to the target task, while the first layers of a CNN can encode more generic feature transformations. A different dataset than the Voxforge corpus would probably be fitter for this kind of transfer learning, as literature and intuition suggests that the closeness of the source to the target task is related to how well the transfer learning works. But especially considering that the Voxforge corpus is recorded in very mixed and varying conditions and that its recording quality is not always very good due to the crowdsourced nature of the project, the obtained results are surprisingly good. We believe that the classification pipeline presented in this paper could also be interesting to other - not only paralinguistic - audio classification tasks.[4]

5. Acknowledgements

The authors would like to thank Steffen Remus, Martin Riedl and Jinseok Nam for insightful discussions and comments.

[4]Source code available at: `https://github.com/bmilde/deepschmatzing`

6. References

[1] B. W. Schuller, "The Computational Paralinguistics Challenge [Social Sciences]," *Signal Processing Magazine, IEEE*, vol. 29, no. 4, pp. 97–101, 2012.

[2] G. L. Trager, "Paralanguage: A first approximation," *Studies in linguistics*, vol. 13, no. 1-2, pp. 1–12, 1958.

[3] B. Schuller, S. Steidl, A. Batliner, F. Burkhardt, L. Devillers, C. Müller, and S. Narayanan, "The INTERSPEECH 2010 Paralinguistic Challenge," *Interspeech*, pp. 2794–2797, 2010.

[4] B. Schuller, S. Steidl, and A. Batliner, "The INTERSPEECH 2009 emotion challenge." in *Interspeech*, 2009, pp. 312–315.

[5] F. Schiel, "Perception of alcoholic intoxication in speech," in *Interspeech*, 2011, pp. 3281–3284.

[6] D. Bone, M. Black, M. Li, A. Metallinou, S. Lee, and S. S. Narayanan, "Intoxicated Speech Detection by Fusion of Speaker Normalized Hierarchical Features and GMM Supervectors." in *Interspeech*, 2011, pp. 3217–3220.

[7] G. E. Hinton, "To recognize shapes, first learn to generate images," *Progress in Brain Research*, vol. 165, pp. 535–547, 2007.

[8] B. Schuller, S. Steidl, A. Batliner, S. Hantke, F. Hönig, J. Orozco-Arroyave, E. Nöth, Y. Zhang, and F. Weniger, "The INTERSPEECH 2015 Computional Paralinguistics Challenge: Nativeness, Parkinson's and Eating Condition," *Interspeech*, 2015.

[9] S. Hantke, F. Weninger, R. Kurle, A. Batliner, and B. Schuller, "I hear you eat and speak: automatic recognition of eating condition and food type," *(to appear)*, 2015.

[10] R. Caruana, "Multitask Learning," *Machine Learning*, vol. 28, pp. 41–75, 1997.

[11] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering (KDE)*, vol. 22, pp. 1345–1359, 2010.

[12] F. Eyben, "openSMILE The Munich Versatile and Fast Open-Source Audio Feature Extractor Categories and Subject Descriptors," *International conference on Multimedia (ACM)*, pp. 1459–1462, 2010.

[13] F. Eyben, F. Weninger, F. Gross, and B. Schuller, "Recent developments in openSMILE, the munich open-source multimedia feature extractor," in *International conference on Multimedia (ACM)*, 2013, pp. 835–838.

[14] I. Lopez-Moreno, J. Gonzalez-Dominguez, O. Plchot, D. Martinez, J. Gonzalez-Rodriguez, and P. Moreno, "Automatic language identification using deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 5337–5341.

[15] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, pp. 255–258, 1995.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2323, 1998.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Neural Information Processing Systems (NIPS)*, pp. 1–9, 2012.

[18] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the Gap to Human-Level Performance in Face Verification," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014, pp. 1701–1708.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," *arXiv preprint*, pp. 1–11, 2015.

[20] T. D. Kulkarni, W. Whitney, P. Kohli, and J. B. Tenenbaum, "Deep convolutional inverse graphics network," *arXiv preprint*, pp. 1–10, 2015.

[21] T. N. Sainath, A. R. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 8614–8618.

[22] H. Lee, P. Pham, Y. Largman, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks." *Neural Information Processing Systems (NIPS)*, pp. 1–9, 2009.

[23] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in *Acoustics, Speech and Signal Processing (ICASSP)*. ICASSP, 2014, pp. 6964–6968.

[24] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint*, pp. 1–14, 2014.

[25] A. E. Hoerl and R. W. Kennard, "Ridge regression: Biased estimation for nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[26] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint*, pp. 1–18, 2012.

[27] P. Simard, D. Steinkraus, and J. Platt, "Best practices for convolutional neural networks applied to visual document analysis," *Seventh International Conference on Document Analysis and Recognition*, p. 958, 2003.

[28] L. Y. Pratt, J. Mostow, C. A. Kamm, and A. A. Kamm, "Direct Transfer of Learned Information Among Neural Networks," in *AAAI*, vol. 91, 1991, pp. 584–589.

[29] L. Y. Pratt, L. Y. Pratt, S. J. Hanson, C. L. Giles, and J. D. Cowan, "Discriminability-based transfer between neural networks," in *Neural Information Processing Systems (NIPS)*, 1993.

[30] A. Sharif, R. Hossein, A. Josephine, S. Stefan, and K. T. H. Royal, "CNN Features off-the-shelf : an Astounding Baseline for Recognition," in *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014, pp. 512–519.

[31] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," *arXiv preprint*, pp. 1–12, 2014.

[32] Y. Nesterov, "A method of solving a convex programming problem with convergence rate O (1/k2)," *Soviet Mathematics Doklady*, vol. 27, pp. 372–376, 1983.

[33] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," *International Conference on Machine Learning (ICML)*, vol. 28, pp. 1139–1147, 2013.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2012.

[35] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math compiler in Python," in *Python in Science Conference (SCIPY)*, 2010, pp. 1–7.