IMPROVING THE INTERPRETABILITY OF DEEP NEURAL NETWORKS WITH STIMULATED LEARNING

Shawn Tan, Khe Chai Sim

National University of Singapore

Mark Gales

University of Cambridge

ABSTRACT

Deep Neural Networks (DNNs) have demonstrated improvements in acoustic modelling for automatic speech recognition. However, they are often used as a black box, and not much is understood about what each of the hidden layers does. We seek to understand how the activations in the hidden layers change with different input, and how we can leverage such knowledge to modify the behaviour of the model. To this end, we propose stimulated deep learning where stimuli are introduced during the DNN training process to influence the behaviour of the hidden units. Specifically, constraints are applied so that the hidden units of each layer will exhibit phone-dependent regional activities when arranged in a 2-dimensional grid. We demonstrate that such constraints are able to yield visible activation regions without compromising the classification of the network and suppressing the activations for a region affects the classification accuracy of the corresponding phone more than the others.

Index Terms— Deep Neural Networks

1. INTRODUCTION

Deep Neural Network (DNN) acoustic modelling has achieved state-of-the-art performance in comparison to the conventional Gaussian Mixture Model (GMM) based systems. A good summary of the techniques used and their results in comparison to GMMs are reported in [1]. Unfortunately, little is understood about what happens in the DNN and how to update models after they have been trained.

DNNs have a generic multi-layer nonlinear structure that makes it powerful and flexible. They are typically learned in a data-driven manner to model complex mapping functions. As such, DNNs are often regarded as a "black box" and the lack of interpretability makes post-training modifications to the model difficult. For Automatic Speech Recognition (ASR), it is important to adapt the DNN-based acoustic model to reduce the mismatch between the development and deployment conditions. Typically, generic approaches such as feature transformation [2] and feature augmentation [3] are used for speaker and noise normalisation. However, these methods focus on improving the classification performance, not interpretability.

The approaches taken in analysing a trained neural network can are usually done by examining (1) the weights in each layer, typically by visualising them in the case of models trained for computer vision tasks, or (2) analysing the activations as transformed representations of the input data. The issues with current approaches is that they seldom focus on being able to make modifications to the trained hidden layers in order to yield different behaviour.

DNNs achieve their predictions through multiple levels of linear transformations and non-linear activations. While a single linear transformation may be interpreted by looking at the weights from the input features to each of the output classes, multiple layers with non-linear interactions at every layer make understanding the DNN a difficult task.

In this paper, we use *stimulated learning*, where we inject information at each of the hidden layers in order to (1) be able to visualise these hidden layers after training, and (2) be able to know with some certainty what a region of the hidden layer is specialised in recognising.

In Section 2, we discuss the existing issues with interpreting neural networks. In Section 3 we will elaborate what *stimulated learning* is, and how it applies to visualising and interpreting neural networks. In Section 4 we explore the model trained using the stimulated deep learning process and discuss some of the properties of the model. Finally, in Section 5 we give an overview of what we have learnt, and future directions this work could move in.

2. NEURAL NETWORK INTERPRETATION

One of the common ways for inspecting the feature importance for neural network models is Garson's algorithm [4], and was improved by [5]. The work proved useful for analysing neural networks, but most of the works in which it is used train neural networks with one single layer. It also seems to be unable to account for the empirical range of the neuron outputs, which we have found in DNNs to be an important factor – when using a sigmoid squashing function, some neurons activate close to 0 or 1.

In the realm of computer vision, analysing neural network weights has been another approach to interpreting neural networks. One of the ways is to treat the inputs as parameters and optimise for a given hidden unit in the network [6, 7, 8]. One

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.

method typically used in interpreting convolutional networks involves inverting the functions and transformations, using signals from the forward propagation step to reconstruct the features detected by a neuron [9]. This works for neural networks for vision since the input is immediately interpretable. This procedure can be replicated with acoustic models, but the resultant spectogram output is hard to interpret.

One paper used t-SNE [10] to visualise the different spaces the intermediate layers of the network and how they transform the original audio frame into the prediction [11]. This approach is useful in trying to get a measure of how the non-linearities of DNNs help with making the input space separable. While this is useful to get an idea of how the feature space is transformed as it goes up the layers, there is no direct way to affect the transformed space to modify the behaviour of the neural network.

3. STIMULATED DEEP LEARNING

The aim of stimulated learning is to be able to augment the training process of the neural network: hidden nodes are stimulated at training time to respond to external attributes (stimuli). Some of the existing work fall under this category, like using DNNs and i-vectors and dropout.

If we can allocate different regions of the hidden layer for learning different concepts, we can then effect change on these known regions of the network to modify its behaviour. In this section, we describe our proposed methods for achieving this. All experiments are carried out using the TIMIT dataset, using fmllr features with a context of 11. The DNN has 6 hidden layers ($L = 6$), and its output is a probability distribution across 1945 senones. In the following experiments, we use neural networks with the same structure with different training targets. All features are extracted using the Kaldi toolkit [12], and the training of the DNNs referenced in the following subsections are trained using the Theano library [13].

Given an input frame $\mathbf{x}$ and a corresponding label y, a DNN computes the hidden vectors $\mathbf{h}_{\mathbf{x}}^{(l)}$,

$$\mathbf{h}_t^{(0)} = \mathbf{x}_t \tag{1}$$

$$\mathbf{h}_t^{(l)} = \sigma(\mathbf{W}^{(l)}\mathbf{h}_t^{(l-1)} + \mathbf{b}^{(l)}), \tag{2}$$

where

$$(\sigma(\mathbf{z}))_i = \frac{1}{1+\exp(-\mathbf{z}_i)}$$

and a probability for the label $P(y|\mathbf{x})$, by the following equations,

$$\mathbf{z} = \mathbf{W}^{(L+1)}\mathbf{h}_t^{(L)} + \mathbf{b}^{(L+1)} \tag{3}$$

$$P(y|\mathbf{x}) = \frac{\exp(\mathbf{z}_y)}{\sum_i \exp(\mathbf{z}_i)} \tag{4}$$

where $l \in \{1, \ldots, L\}$.

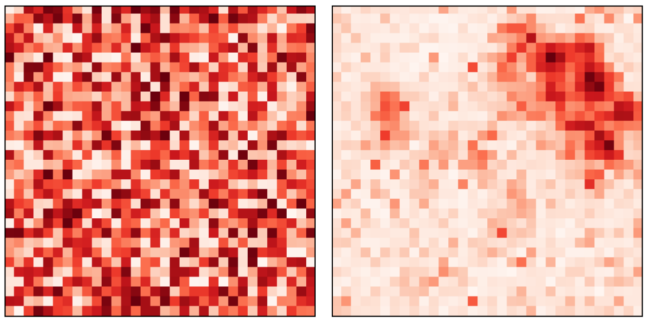

(a) Unstimulated (b) Stimulated

Fig. 1. Unstimulated and stimulated network

We seek to understand and visualise $\mathbf{h}_t^{(l)}$ better, and in the following subsections, we propose a way of introducing stimuli in the training process in order to improve the hidden layers' interpretability.

3.1. Arbitrary Ordering Problem

An issue with the interpretation of the hidden layers in DNNs is that the cells in those layers are arbitrarily ordered. This presents two problems, (1) visualising these hidden layers can give us no insights and (2) the lack of a spatial ordering restricts us to manipulating these neurons as independent items instead of groups of similarly functioning neurons. One example of this is in Learning Hidden Unit Contribution (LHUC) [14] where the number of parameters to be adapted is equal to the number of neurons in the hidden layer. This is because each neuron has to be treated independently as there is no notion of similarity between nearby neurons. As an example, Figure 1 shows the activations of 1024 hidden units, arranged in a 32 by 32 grid, for two DNNs. The left plot corresponds to an unstimulated network where the hidden layer activations appear to be random and difficult to visualise. The right plot corresponds to a stimulated network where active hidden units form a cluster that is much easier to visualise and interpret.

Some applications in vision have imposed a spatial ordering to weights for their hidden layer neurons: feature detectors which detect similar features are arranged closely on a 2-dimensional grid [15]. This facilitates another level of pooling later where the feature detectors are pooled for a higher level of abstraction. This is achieved in part by adding an additional regularisation term to the cost function in order to reduce differences between groups of feature detectors.

Similarly, we can impose a penalty on the cost function to prefer activations to be grouped in a region. We propose one method for stimulating the nodes in the hidden layers of the DNN, and impose that as a penalty in the optimisation cost.

First, we define some d-dimensional space in which each

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.

neuron on layer l exists. This can be described by a (k_l, d) matrix $\mathbf{S}$. We can then impose a penalty for a given d-dimensional point $\mathbf{s}$ to ensure activations are higher in that area. One method for achieving this is to impose a Gaussian constraint centred around that point,

$$\hat{g}(i;\mathbf{s}) = \exp\left(-\frac{1}{2}\|(\mathbf{S}_i)^\top - \mathbf{s}\|^2\right) \tag{5}$$

$$g(i;\mathbf{s}) = \frac{\hat{g}(i;\mathbf{s})}{\sum_j^{k_l} \hat{g}(j;\mathbf{s})} \tag{6}$$

$g(\cdot;\mathbf{s})$ defines a surface with which we want the hidden layer contributions to conform to and $\mathbf{S}_i$ is the ith row of $\mathbf{S}$. We penalise the contribution differences using the KL-divergence cost,

$$D(\mathbf{h}, \mathbf{s}) = \sum_i g(i;\mathbf{s}) \cdot \log \frac{g(i;\mathbf{s})}{\mathbf{h'}_i}, \tag{7}$$

$$\mathbf{h'}_i = \frac{\mathbf{h}_i}{\sum_j \mathbf{h}_j}, \tag{8}$$

These penalties are then added to our cost function depending on how the constraint is to be applied. In our experiments, we apply this additional cost to all layers, using the same $\mathbf{S}$ throughout, which gives us similar regions activating across layers when we feed in an input vector.

3.2. Imposing the penalty

In addition to the standard cross-entropy criteria for training, we also stimulate the network using the ordering penalty over the contribution metric we discussed in Section 2.

$$L = \frac{1}{T}\sum_t^T \left(-\log P(y_t|\mathbf{x}_t) + \alpha \sum_l^L D(\mathbf{h}_t^{(l)}, \mathbf{s}_{p_t})\right)$$

where α is the weight given to the importance of the penalty term, and $\mathbf{s}_{p_t}$ is the 2-dimensional point for the corresponding phoneme at time t.

This constraint jointly optimises: 1) for the neural network to satisfy both the cross-entropy cost with the label y_t, and 2) the positioning stimuli for all layers as a secondary constraint. All layers are stimulated with the same t-SNE mapping (As seen in Figure 3).

The importance of the secondary constraint is controlled by α. Figure 2 shows the final cross-entropy values against the final KL-divergence values on the validation set. Higher values of α results in lower values of the secondary constraint. The cross-entropy objective initially improves, but then increases in value with higher values of α. This is likely due to the regularisation effect of the additional constraint, allowing the model to generalise better. In the following analyses, we use the model trained with $\alpha = 0.1$ which achieves a PER of 19.3%, while a standard model ($\alpha = 0.0$) has a PER of 19.4%. This demonstrates that we are able to perform the stimulated learning with no degradation in model accuracy.

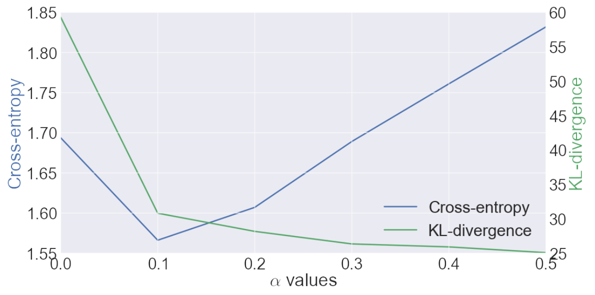


Fig. 2. Trade-off between KL-divergence penalties and cross-entropy costs on the validation set given different values of α.

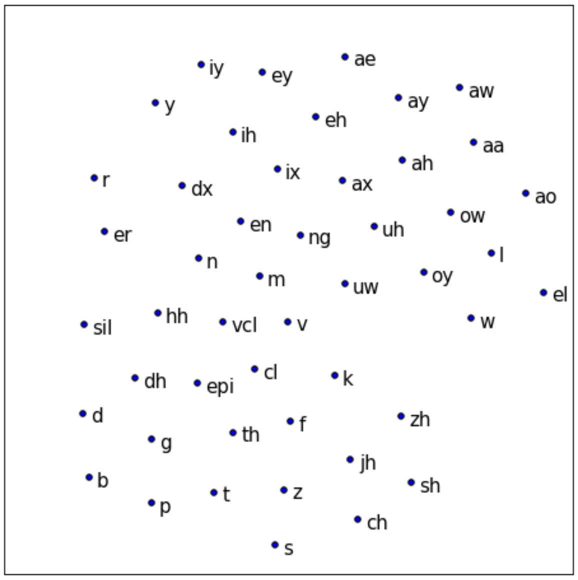


Fig. 3. 2-dimensional mapping of phonemes.

3.3. Defining points for stimulation

We can define a point for each phoneme to centre activations around, stimulating the hidden layer to light up around those regions when a frame of that phoneme is given as input.

These points were obtained by applying t-SNE on the average fmllr frame of each phoneme. The resulting 2-dimensional points were then centred with the mean, and then scaled to fit within the 32 by 32 boundary. Figure 3 is the plot of the resulting points.

The use of t-SNE ensures that similar types of phonemes are positioned close together in the 2-dimensional space that we are reducing it down to. This is to allow for regions of activations to be more contiguous if there is a confusion between two or more similar phonemes.

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.

3.4. Arbitrary Scaling Problem

The hidden activations are sometimes referred to as 'squashing' functions, and this is because they restrict the values to within a range. In the case of the sigmoid function, this range is (0,1). In practice, however, the range of the units do not extend that full range of values. When analysing the hidden layers, there are neurons that stay close to the extreme ends of that range.

It is easy to dismiss these nodes as being useless, particularly if they do not seem to vary much from input to input. Unfortunately, the contribution they make to the activations they make at the next layer also depend on the outgoing weights for that neuron. Conversely, we cannot conclude that a neuron is important by looking at its outgoing weights, when the range of its activations should also be a factor.

We propose a different metric that considers both weights and activations. Given the l-th hidden layer $\mathbf{h}_t^{(l)}$, we consider the weights $\mathbf{W}^{(l+1)}$, the outgoing connections from $\mathbf{h}_t^{(l)}$. Then the contributions to the next layer $\mathbf{h}_t^{(l)}$ for an input frame $\mathbf{x}_t$ is defined as

$$\left(\mathbf{h}_t^{(l)}\right)_i = \left(\mathbf{h}_t^{(l)}\right)_i \cdot \sqrt{\sum_j \left(\mathbf{W}^{(l+1)}\right)_{ji}^{2}},$$

where $(\cdot)_i$ is the i-th element of the vector. In other words, we use the norm of the outgoing connections as the measure contribution from one layer to the next. This gives us a way to take both aspects of the problem into account, the empirical range of the neuron's activation, and the importance the next layer's weights assigns to it.

4. HIDDEN LAYER ANALYSIS

We can now inspect the hidden layers by forward propagating frames and plotting the hidden layers on a 32 by 32 pixel image. Figure 6 shows the activations across all 6 layers in the DNN at particular points during the utterance. In particular, we have highlighted 5 different parts of the utterance and shown the activation plots for all 6 layers of the DNN.

It is clear that the constraints have achieved the goal of concentrating activations close together. The phoneme `sil` is the best phoneme when ranked in ascending KL-divergence cost order, and this can also be seen in the plots as cleaner and smoother activations. The level of spurious activations in the layer also centre around the t-SNE projected point generally decreases until the fourth layer, and then increases for the fifth and the sixth. The KL-divergence values per layer also shows a similar trend. The vowel phonemes are in the top right hand corner of the plot, and these seem to visually have the largest spread. In the Figure 6, this phenomenon can be seen for the plots in `ow` and `iy`.

When comparing the plots for `v` and `s` with the positions in Figure 3, we can also observe the sensible mistakes that the network makes. In the initial layers, there is a spread from the `s` point toward that of the nearby `z` point. The `v` activations also has a spread towards the `th` and `f` sounds.

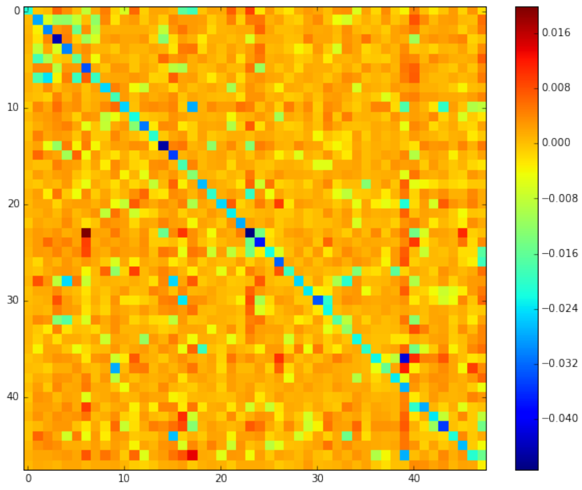


Fig. 5. Plot of the relative accuracy reduction when muting out regions of the hidden layers

4.1. Modifying behaviour using known regions

We now know regions in the hidden layers that correspond to phonemes. We mute these regions to introduce a *handicap* to the model and observe the classification accuracy on the development set of TIMIT.

We do this by multiplying a Gaussian mask with the hidden layer, redefining $\mathbf{h}_t^{(l)}$ as,

$$(\mathbf{h}_t^{(l)})_i := (1 - \hat{g}(i; \mathbf{s}_p)) \cdot (\mathbf{h}_t^{(l)})_i$$

This essentially masks away the region that is expected to activate for phoneme p. We then forward propagate using this modified version of the neural network and observe the accuracy for the senones associated with each phoneme.

Figure 5 shows the relative accuracy across each phoneme. The rows represent the phoneme regions that were muted during testing, and the columns represent the accuracy of the phonemes in relation to the default (un-muted) accuracy for that phoneme. The relative difference seen in the diagonal of the plot implies that the regions being muted more strongly affect the corresponding phoneme that we are deciding to handicap.

This suggests that the stimulation we impose during the training do not only give a more structured visual representation of the hidden activations, but also cause the DNN to be learnt in a way that common concepts are grouped in the hidden layers, to the extent that this can be manipulated at test time to cause changes in the behaviour of the DNN.

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.

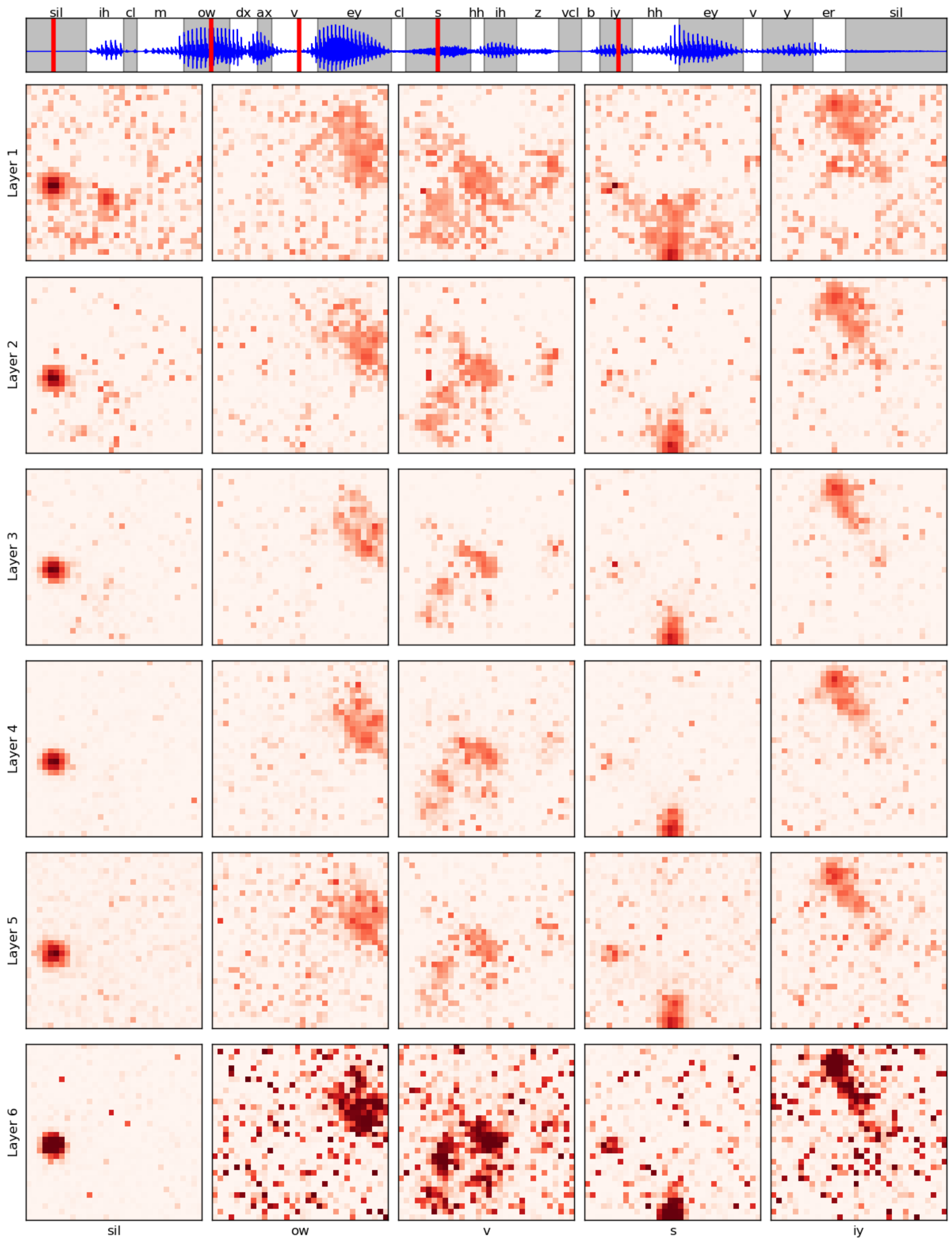


Fig. 4. Hidden layer plots across points in an utterance.

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.

4.2. Training with contextual information

We wanted to know if the network could generalise to stimuli that was unseen in the training data. We trained another network with contextual information: the left and right context phoneme of the current phoneme.

We used a mixture of Gaussians as above, with the surfaces normalised to 1. The same KL-divergence cost is used for each layer, with $\alpha = 0.1$. We attain a model with a PER of 19.2%, demonstrating again that we are able to do this without degradation of the original model.

Figure 6 shows from left to right plots of hidden layer activations for (a) seen contexts and (b) unseen contexts. In training this model, we find that the same trend, with KL-divergence costs decreasing in the first four layers and then increasing in the fifth and sixth does not hold here. In addition, the unseen contexts get confused with more common contexts that are usually seen, and do not generalise like we would expect. Visually, we see that the activations are more 'messy' in the first layer, and then clean up later on in the higher layers. This suggests that the model does some abstraction as it goes up the layers that is useful in determining what the context of the window of audio frames are.

5. SUMMARY AND FUTURE WORK

In this paper, we have addressed two main problems when trying to analyse the hidden layers of DNNs, and proposed a way of stimulating the training of DNNs in order to address these problems.

We demonstrate that the stimuli produce the intended results of creating activities in the hidden layers that conform to the constraints we introduce. Setting low α values for the constraint also help to improve the decoding performance of the acoustic model. Additionally, we show that we can use this knowledge of where the activations are for each phoneme to introduce a phoneme specific handicap. These handicaps were then shown to specifically target only the affected phoneme, which demonstrates the capability to modify the behaviour of these stimulated DNNs. We were however unable to use contextual information as stimuli and have the hidden layers generalise over contexts that were unseen in the training data.

The long-term goal of this work is to (1) better understand, and improve the interpretability of neural networks through different training methods, and (2) to be able to leverage that to modify and improve trained networks. Our work has demonstrated that it is possible to stimulate the hidden layers during the training process in order to gain a more interpretable representation, but future work will include applying these techniques for post-training modifications to the network, such as speaker adaptation or noise adaptation.

6. ACKNOWLEDGEMENTS

This research was supported by Singapore Ministry of Education Academic Research Fund Tier 2 (Official Project No: MOE2014-T2-1-068).

7. REFERENCES

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[2] Frank Seide, Gang Li, Xie Chen, and Dong Yu, "Feature engineering in context-dependent deep neural networks for conversational speech transcription," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011, pp. 24–29.

[3] A. Senior and I. L. Moreno, "Improving dnn speaker independence with i-vector inputs," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014.

[4] G. David Garson, "Interpreting neural-network connection weights," *AI Expert*, vol. 6, no. 4, pp. 46–51, Apr. 1991.

[5] ATC Goh, "Back-propagation neural networks for modeling complex systems," *Artificial Intelligence in Engineering*, vol. 9, no. 3, pp. 143–151, 1995.

[6] Anh Nguyen, Jason Yosinski, and Jeff Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," *arXiv preprint arXiv:1412.1897*, 2014.

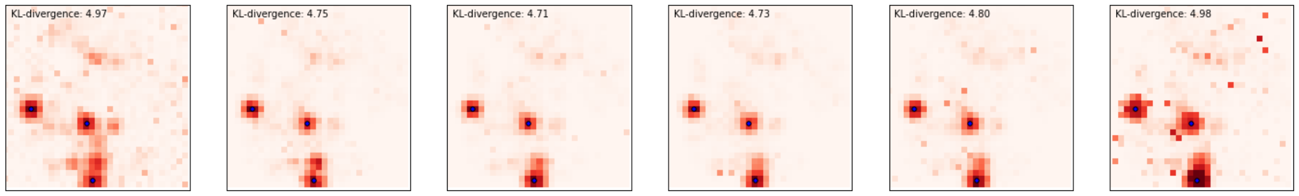
[7] Aravindh Mahendran and Andrea Vedaldi, "Understanding deep image representations by inverting them," *arXiv preprint arXiv:1412.0035*, 2014.

[8] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

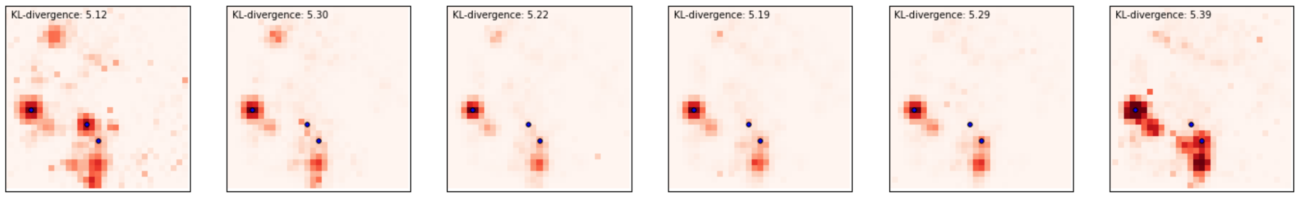
[9] Matthew D Zeiler and Rob Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision–ECCV 2014*, pp. 818–833. Springer, 2014.

[10] Laurens Van der Maaten and Geoffrey Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, pp. 85, 2008.

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.



(a) Seen contexts



(a) Unseen contexts

Fig. 6. Example of hidden layer activations for a seen context and an unseen context

[11] Abdel Rahman Mohamed, Geoffrey Hinton, and Gerald Penn, "Understanding how deep belief networks perform acoustic modelling," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2012.

[12] D. Povey, a. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2011, pp. 1–4.

[13] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio, "Theano: A cpu and gpu math compiler in python," in *Proc. 9th Python in Science Conf*, 2010, pp. 1–7.

[14] Pawel Swietojanski and Steve Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 2014, pp. 171–176.

[15] Koray Kavukcuoglu, Marc Aurelio Ranzato, Rob Fergus, and Yann Le-Cun, "Learning invariant features through topographic filter maps," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 1605–1612.

Authorized licensed use limited to: NATIONAL UNIVERSITY OF IRELAND GALWAY. Downloaded on March 28,2021 at 09:44:57 UTC from IEEE Xplore. Restrictions apply.