International Workshop on Machine Learning in Visualisation for Big Data (2020)
D. Archambault, I. Nabney and J. Peltonen (Editors)

Visual Interpretation of DNN-based Acoustic Models using Deep Autoencoders

Tamás Grósz [†] and Mikko Kurimo

Department of Signal Processing and Acoustics, Aalto University, Finland

Abstract

In the past few years, Deep Neural Networks (DNN) have become the state-of-the-art solution in several areas, including automatic speech recognition (ASR), unfortunately, they are generally viewed as black boxes. Recently, this started to change as researchers have dedicated much effort into interpreting their behavior. In this work, we concentrate on visual interpretation by depicting the hidden activation vectors of the DNN, and propose the usage of deep Autoencoders (DAE) to transform these hidden representations for inspection. We use multiple metrics to compare our approach with other, widely-used algorithms and the results show that our approach is quite competitive. The main advantage of using Autoencoders over the existing ones is that after the training phase, it applies a fixed transformation that can be used to visualize any hidden activation vector without any further optimization, which is not true for the other methods.

CCS Concepts

• ***Computing methodologies*** $\rightarrow$ ***Dimensionality reduction and manifold learning;*** *Speech recognition; Neural networks;*

1. Introduction

Using Deep Neural Networks (DNN) has become a common practice in automatic speech recognition (ASR) since they can achieve the best accuracy [HDY*12]. With the widespread usage of DNNs it is crucial that we develop tools and algorithms so that users can understand when a model works correctly, and when it fails [HKPC18]. Unfortunately, the DNNs inner workings are still a mystery, and it is extremely hard to translate their function into an understandable format for humans. Interpretation is also essential in verifying that a highly accurate DNN has actually learned to use a proper problem representation, and not just exploited some artifacts present in the training data.

Visual analytics in deep learning is a new and very important area. Within this field, we focus on visually explaining how our DNN-based acoustic model functions, specifically, we wanted to see which phonetic categories were recognized by it. The task of visual interpretation is quite simple; given some test data, the hidden representations of the network are inspected [Lip18]. To achieve this, the hidden activation vectors of the DNN need to be transformed into a low dimensional space, once this has been done, humans can visually inspect them in the hope of determining qualitatively what the model has learned. The main advantage of this concept of interpretability is that we can inspect the models, without modifying them. This means that there is no need to sacrifice predictive performance by adding a bottleneck layer to the DNN just to improve its interpretability. Here, we compared methods that can show the phonetic categories, which were recognized by the DNN. Currently, several solutions exist that can be used for the visual interpretation of DNNs, in [HKPC18] we can find an extensive survey of them. Here, we used T-Distributed Stochastic Neighbor Embedding (t-SNE) [vdMH08], Uniform Manifold Approximation and Projection (UMAP) [MH18]. They all focus on transferring as much of the structural information from the high-dimensional data to a lower-dimensional space as possible.

One of the first articles that dealt with interpreting DNNs in ASR, investigated why their method works so well and visualized the similarity structure of the input and the hidden activity vectors using t-SNE [MHP12]. Later, in [VWS14] the authors investigated how a multilingual bottleneck affects the learning process, and again t-SNE was used to display the learned multilingual features. Not long ago, Nagamine et al. studied how DNNs form phonetic categories [NSM15]. Similarly, in [BWJR18] it was inspected how a DNN with a bottleneck layer learns phonetic information by using linear discriminant analysis (LDA) and t-SNE to visualize the hidden activations of a small bottleneck layer of an acoustic DNN. In the literature, it is quite common that a bottleneck layer is used to assist the interpretation [BJRW15, WBR*16, LKH15]. UMAP has also been used to visualize data in different domains [MH18,BMH*19]. The main drawback of t-SNE and UMAP is that they rely on the optimization step that determines the best low-dimensional layout. One problematic consequence is that after

[†] Tamás Grósz was supported by the Kone Foundation.

DOI: 10.2312/mlvis.20201103

any modification of the data (adding or removing some vectors), the optimization must be performed again, which might produce a very different transformation. A further problem is that both t-SNE and UMAP optimize the layout of the whole data globally, severely limiting the number of vectors that we can inspect with them.To solve these issues, we propose the general use of deep Autoencoders (AE) [Bal12]. AEs are well-known and widely used tools for dimension reduction as they can extract meaningful latent features [KH11, VLL*10]. An AE is trained to generate the same data as the one it received on the input layer. It consists of an encoder and a decoder, and the two are connected through a hidden layer.

In this paper, the AE-based method is compared with two standard algorithms, namely with t-SNE and UMAP. Even though we used these methods to inspect acoustic DNNs, they are general methods and can be used to visualize the hidden representations of any network. Naturally, it is hard to compare these methods as they optimize different losses. Previously, Procrustes Distance (PD) [Ken89] was used to compare UMAP with t-SNE, and it was shown that it can measure the stability of the overall structure of the embedding [MH18]. Furthermore, to ensure that we compare the approaches appropriately, we also calculated Mutual Information (MI) [CT91] and Distance Correlation (DC) [SR09] between the data and its visualization to determine the quality of the embeddings. The results indicate that AEs are applicable for visual interpretation. Furthermore, once trained, the learned transformation (the encoder part) can be reused to visualize new data, without any optimization, which is the main advantage of our system over the others.

2. Methods for visualizing the hidden representations

There are many ways of explaining the behavior of a DNN. In this work, we focus on visualization techniques that can be used to explain the decisions of the networks. The main goal of these approaches is to show us what happens inside the network. Usually, it is achieved by transforming the outputs of the hidden layers into a two-dimensional space so that they can be examined. Finding an appropriate transformation is quite difficult, as it needs to take into account that similar vectors in the original high-dimensional space must have projections close to each other in the embedded space.

2.1. T-Distributed Stochastic Neighbor Embedding

Perhaps the best-known option is t-SNE, which was proposed as a dimension reduction method in 2008 [vdMH08]. Since then, it has become a widely used tool for visually interpreting DNNs. The algorithm has two main steps; first, it calculates a similarity measure between the points in the original space, then using an optimizer method, it places them into the embedded space. In the first step, t-SNE calculates a conditional probability for each pair of points that reflects how likely it is that one chooses the other as its nearest neighbor. The algorithm calculates pairwise distances (d) using these probabilities, then the vectors are projected into a low dimensional space by some initial transformation. Afterward, we measure the pairwise distances of the low dimensional vectors (q), and to get the best layout, t-SNE optimizes the transformation by minimizing the Kullback-Leibler divergence between d and q. In recent years, T-SNE has been widely used to interpret DNNs trained for image processing [EKN*17], natural language processing [NKB15], and speech recognition [BWJR18].

2.2. Uniform Manifold Approximation and Projection

The Uniform Manifold Approximation and Projection (UMAP) method [MH18] is a recently proposed manifold learning technique for dimension reduction. The first step of the algorithm is to find a Riemannian manifold on which the data-points are uniformly distributed. The requirement of uniformity is quite problematic since real data sets rarely behave like this. Fortunately, with a Riemannian metric that is not inherited from the ambient space, we can achieve this. In the next step, a fuzzy topological representation needs to be constructed, to achieve this the algorithm patches together the local fuzzy simplicial set representations of local manifold approximations. The same algorithm is used to calculate the fuzzy topological representation of the initial low-dimensional representations too. In the optimization phase, UMAP modifies the layout of the embeddings to minimize the cross-entropy between the two topological representations (the original one and the low-dimensional one). Unlike t-SNE, UMAP can be fitted on some training data and used later on other test data, without approximating a new manifold. Still, an optimization step needs to be performed on the test data.

2.3. Autoencoders

As an alternative, we propose the use of AEs [Bal12] to perform the transformation of hidden representations into a 2D space for visual inspection. AEs are a very special type of Neural Networks, they are trained in a self-supervised manner, and their most important property is that their expected output is the same as their input. These networks aim to learn some meaningful representation of the data. The network is split into two parts; the encoder part, which learns to extract the compact features, while the decoder part is used to reconstruct the input from these features. The activation values of the hidden layer located at the intersection of the encoder and decoder parts will form the new feature vector.

AEs are known to produce representations that retain the structure of the original data. This is why they are widely used for dimension reduction tasks, such as word embedding. Naturally, the question arises; how do AEs preserve the local structure of the data? A recent article has already tackled this question by using an advanced information-theoretic methodology to understand the dynamics of learning and the design of AEs [YP19]. They extend the Data Processing Inequality [BR12] to deep AEs and show that if the structure is symmetric and a linear activation is used in the bottleneck layer, then the role of the AE is to maximize the entropy in the hidden layers (if it is well trained with backpropagation). This explains why the latent vectors, extracted by the encoder, represents the original data so well.

For visualization a special network structure can be used in which the feature extractor layer has only two neurons. In our experiments, we opted for a special architecture; each hidden layer reduces the dimension, first from 1000 to 100, then to 10, and finally, the last hidden layer in the encoder has two neurons. The decoder is symmetric to the encoder. We trained the AEs by minimizing the mean squared error (MSE) between its input and output. After training, only the encoder part is needed to transform new vectors, and no further optimization is needed. As a result of this, AEs are much faster than the other methods, and they produce the same embedding for a given vector, regardless of what other data we use during the visualization step.

3. Experimental setup

The DNN interpreted here is a fully connected rectifier network [GBB11] that has five hidden layers, each containing 1000 hidden neurons. It was trained as a phoneme recognizer on the Wall Street Journal corpus (si-284 set) [PB92] using a GMM-free algorithm [GGT17].For visualization, we randomly choose three files from the test set (these files will be referred to as the test data).

The AEs were trained with TensorFlow [AAB*15]. As input, they received the hidden state vectors calculated by using the training data. We trained one AE for each inspected layer. The other methods performed the optimization step using the test data. The only exception was the UMAP method, which provided a way of estimating the manifold using some training data. Due to the memory limitations, we could not use all the training data, so we randomly selected 300 training utterances. Please note that the optimization step was still performed using the three test files.

During visual interpretation, we checked how well the main phone categories (silence, vowel, semivowels, and consonants) were separated from each other. Then, we inspected the consonants to show which subcategories were recognized by the hidden layers.

3.1. Evaluation metrics

Since all three algorithms optimize different metrics to perform the transformation, we need metrics that can be used to compare the performances objectively. Here, we used three different metrics as there is no established metric for this task.

Procrustes Distance (PD) can be used to match two configurations of points and measure their similarities. The embeddings and the high dimensional data are used to find an optimal transformation (T) between them. Let us denote the original points by X and their embedding by Y. The algorithm optimizes T by minimizing the mean squared error (MSE) between X and TY. Once the optimization is done, PD is defined as the MSE of the best T.

Mutual Information (MI) is a standard metric, which measures the mutual dependence between two variables. Formally, it is defined as $MI(X,Y) = E(X) + E(Y) - E(X,Y)$, where E is the entropy function. As the exact probability distributions were not available, the Kraskov-Stögbauer-Grassberger method (KSG) [KSG04] was utilized to estimate the entropy. KSG estimates are based on the distances between k nearest neighbors (here, we used $k = 10$).

Distance Correlation (DC) [SR09] is a metric that measures the correlation between paired vectors of arbitrary dimension. First, it calculates the pairwise distances between the vectors, separately for the two sets. Then, DC is defined as the correlation between pairwise distances.

Table 1: *Results got by applying different visualization methods*

layer	metric	t-SNE	UMAP	UMAP-train	AE
1	PD	0.909	0.919	0.923	**0.896**
	MI	0.341	**0.460**	0.429	0.392
	DC	0.755	0.823	**0.828**	0.793
2	PD	0.936	0.948	0.939	**0.928**
	MI	0.402	0.470	**0.559**	0.544
	DC	0.640	0.813	0.838	**0.871**
3	PD	0.970	0.960	0.955	**0.948**
	MI	0.407	0.485	0.453	**0.503**
	DC	0.497	0.804	0.807	**0.882**
4	PD	0.976	0.968	0.965	**0.952**
	MI	0.499	**0.586**	0.491	0.495
	DC	0.431	0.797	0.804	**0.871**
5	PD	0.977	0.968	0.966	**0.938**
	MI	0.433	0.497	0.502	**0.518**
	DC	0.461	**0.805**	0.796	0.791

Regarding these metrics, one might argue that PD favors the AE-based approach as it measures how well the original data can be reconstructed from the embeddings. Meanwhile, MI focuses on the preservation of the local structures, which is the strength of both t-SNE and UMAP. Lastly, DC measures the similarity between the global structure of the original and projected data.

4. Results

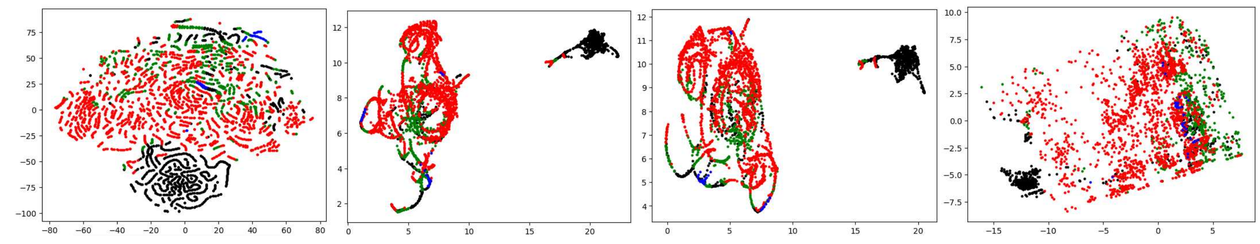
We tested four different visualization approaches. UMAP denotes the embeddings that we created using just the test data and UMAP-train was first fitted using the 300 utterances selected from the train set and then optimized on the test data. Table 1 shows the results of all four approaches. Keep in mind that AEs were not trained on the test data, while the other systems were explicitly optimized on it.

Based on the PD values, AEs produced the best results, and UMAP outperformed t-SNE in most cases. We also calculated the PD values of random coordinates; on average, it was 0.99, slightly worse than those of the four methods, mainly because of the optimization step in the PD calculation. This indicates that PD might not be a good choice to measure the quality of embeddings. In terms of MI, we can see that the UMAP-based methods achieved the best results, except for the third layer, and AEs came second, meaning that it managed to keep a sufficient amount of information. Interestingly, using more data to estimate the manifold (UMAP-train) is not always beneficial. Observing the DC values, we can say that AE and the UMAP methods perform well, while t-SNE produces markedly worse results.

4.1. Visual interpretation

After the objective comparison, we inspected the embeddings visually. Figure 1 shows the outputs of the four approaches for the first hidden layer, which processed the input directly. The silent parts were separated from the other categories by the first layer, and it also started to distinguish vowels from consonants to some degree.

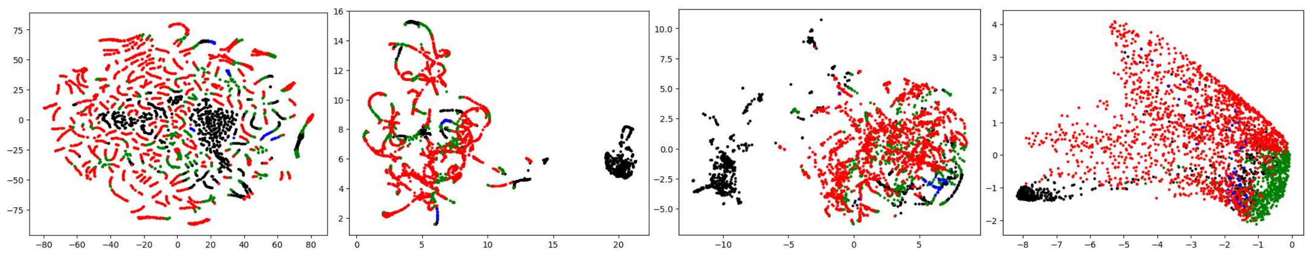
Another important part of a DNN is the last hidden layer, as



(a) *t-SNE* **(b)** *UMAP* **(c)** *UMAP-train* **(d)** *Autoencoder*

● Silence, ● Consonants, ● Semivowels, ● Vowels

Figure 1: *Visual representations of the activation vectors outputted by the first hidden layer that processes the input directly. Coordinates are determined by the visualization method and color is assigned to each point based on the force aligned labels.*



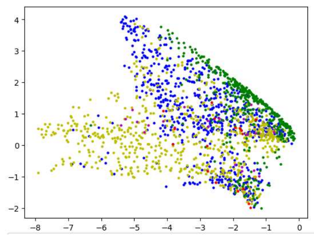
(a) *t-SNE* **(b)** *UMAP* **(c)** *UMAP-train* **(d)** *Autoencoder*

● Silence, ● Consonants, ● Semivowels, ● Vowels

Figure 2: *Visual representations of the activation vectors outputted by the final hidden layer.*

the output layer makes decisions based on the output of this layer. In this case (see Figure 2) we once again notice that the silence parts were recognized accurately. Here, we would like to point out that UMAP and UMAP-train visualize the same test data, the only difference being that UMAP-train utilizes some additional data in the manifold approximation step. In theory, these images should be very similar. However, as can be observed, they are quite different as a result of data change. From the AE-based image, it is clear that consonants were isolated from vowels, but the other methods do not reinforce this observation. One possible reason for this might be that the other methods did not use enough data. To test our hypothesis, we generated new visualizations by using the 303 utterances during the optimization step of UMAP. This new image showed vowel and consonant groups formed similarly to the AE. Based on this observation, we can say that using more data gives more accurate visualization, at the cost of increased processing time. Inspecting the vowels, we saw that no groups have formed, which means that the DNN did not differentiate between the vowel categories.

Lastly, we checked the consonants to see if any other structure was learned by the DNN. Figure 3 tells us that nasal phones formed a separate group, and stop phones and fricative ones were also separated to some degree. These effects could be observed when using AEs. On the other hand, t-SNE did not show such separation; neither did UMAP when we used only the test data. Once we used more data (303 utterances) to perform the UMAP embeddings, the observation that the stop, nasal and fricative categories are separated was confirmed. Based on this, we can assume that the last hidden layer learned to recognize nasal, fricative, and stop consonants as a side effect of the phoneme classification task.



● Stop, ● Fricative, ● Affricative, ● Nasal, ● Aspirate

Figure 3: *Embeddings of consonants produced by autoencoder*

5. Conclusions

In this study, we showed that DAEs could be used to visually interpret DNNs. We compared our approach with two other popular methods and found that it is competitive, even though it does not optimize the layout of the data that we want to visualize. A great advantage of the AE-based visualization is that we can train them with a large amount of data, and after training, the encoder applies a fixed transformation to produce the coordinates. This also means that the visualization will not be affected by artifacts present only in the data that we wish to inspect as the layout is not optimized using this data.

References

[AAB*15] ABADI M., AGARWAL A., BARHAM P., BREVDO E., CHEN Z., CITRO C., CORRADO G. S., DAVIS A., DEAN J., DEVIN M., GHEMAWAT S., GOODFELLOW I., HARP A., IRVING G., ISARD M., JIA Y., JOZEFOWICZ R., KAISER L., KUDLUR M., LEVENBERG J., MANÉ D., MONGA R., MOORE S., MURRAY D., OLAH C., SCHUSTER M., SHLENS J., STEINER B., SUTSKEVER I., TALWAR K., TUCKER P., VANHOUCKE V., VASUDEVAN V., VIÉGAS F., VINYALS O., WARDEN P., WATTENBERG M., WICKE M., YU Y., ZHENG X.: TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 3

[Bal12] BALDI P.: Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning* (02 Jul 2012), vol. 27 of *Proceedings of Machine Learning Research*, PMLR, pp. 37–49. 2

[BJRW15] BAI L., JANCOVIC P., RUSSELL M., WEBER P.: Analysis of a low-dimensional bottleneck neural network representation of speech for modelling speech dynamics. In *Proc. Interspeech* (09 2015). 1

[BMH*19] BECHT E., MCINNES L., HEALY J., DUTERTRE C.-A., KWOK I. W., NG L. G., GINHOUX F., NEWELL E. W.: Dimensionality reduction for visualizing single-cell data using umap. *Nature biotechnology 37*, 1 (2019), 38. 1

[BR12] BEAUDRY N. J., RENNER R.: An intuitive proof of the data processing inequality. *Quantum Info. Comput. 12*, 5-6 (May 2012), 432–441. URL: http://dl.acm.org/citation.cfm?id=2230996.2231000. 2

[BWJR18] BAI L., WEBER P., JANČOVIČ P., RUSSELL M.: Exploring how phone classification neural networks learn phonetic information by visualising and interpreting bottleneck features. In *Proc. Interspeech* (2018), pp. 1472–1476. 1, 2

[CT91] COVER T. M., THOMAS J. A.: *Elements of Information Theory*. Wiley-Interscience, New York, NY, USA, 1991. 2

[EKN*17] ESTEVA A., KUPREL B., NOVOA R. A., KO J., SWETTER S. M., BLAU H. M., THRUN S.: Dermatologist-level classification of skin cancer with deep neural networks. *Nature 542*, 7639 (2017), 115. 2

[GBB11] GLOROT X., BORDES A., BENGIO Y.: Deep Sparse Rectifier Neural Networks. In *Proc. of the International Conference on Artificial Intelligence and Statistics (AISTATS)* (Ft. Lauderdale, FL, USA, 2011), Gordon G. J., Dunson D. B., (Eds.), vol. 15, Journal of Machine Learning Research - Workshop and Conference Proceedings, pp. 315–323. 3

[GGT17] GRÓSZ T., GOSZTOLYA G., TÓTH L.: A Comparative Evaluation of GMM-Free State Tying Methods for ASR. In *Proc. Interspeech* (2017), pp. 1626–1630. doi:10.21437/Interspeech.2017-899. 3

[HDY*12] HINTON G., DENG L., YU D., DAHL G. E., MOHAMED A.-R., JAITLY N., SENIOR A., VANHOUCKE V., NGUYEN P., SAINATH T. N., ET AL.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine 29*, 6 (2012), 82–97. 1

[HKPC18] HOHMAN F., KAHNG M., PIENTA R., CHAU D. H.: Visual analytics in deep learning: An interrogative survey for the next frontiers. *IEEE Transactions on Visualization and Computer Graphics 25* (2018), 2674–2693. 1

[Ken89] KENDALL D. G.: A survey of the statistical theory of shape. *Statistical Science 4*, 2 (1989), 87–99. 2

[KH11] KRIZHEVSKY A., HINTON G. E.: Using very deep autoencoders for content-based image retrieval. In *Proc. ESANN* (2011). 2

[KSG04] KRASKOV A., STÖGBAUER H., GRASSBERGER P.: Estimating mutual information. *Phys. Rev. E 69* (Jun 2004), 066138. URL: https://link.aps.org/doi/10.1103/PhysRevE.69.066138, doi:10.1103/PhysRevE.69.066138. 3

[Lip18] LIPTON Z. C.: The mythos of model interpretability. *ACM Queue 16*, 3 (June 2018), 30:31–30:57. doi:10.1145/3236386.3241340. 1

[LKH15] LIU Y., KARANASOU P., HAIN T.: An investigation into speaker informed dnn front-end for lvcsr. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2015), IEEE, pp. 4300–4304. 1

[MH18] MCINNES L., HEALY J.: Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426* (2018). 1, 2

[MHP12] MOHAMED A., HINTON G., PENN G.: Understanding how deep belief networks perform acoustic modelling. In *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (March 2012), pp. 4273–4276. 1

[NKB15] NARASIMHAN K., KULKARNI T., BARZILAY R.: Language understanding for text-based games using deep reinforcement learning. In *Proc. of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015), Association for Computational Linguistics, pp. 1–11. doi:10.18653/v1/D15-1001. 2

[NSM15] NAGAMINE T., SELTZER M. L., MESGARANI N.: Exploring how deep neural networks form phonemic categories. In *Proc. Interspeech* (2015). 1

[PB92] PAUL D. B., BAKER J. M.: The Design for the Wall Street Journal-based CSR Corpus. In *Proc. of the Workshop on Speech and Natural Language* (Stroudsburg, PA, USA, 1992), HLT '91, Association for Computational Linguistics, pp. 357–362. doi:10.3115/1075527.1075614. 3

[SR09] SZÉKELY G. J., RIZZO M. L.: Brownian distance covariance. *Ann. Appl. Stat. 3*, 4 (12 2009), 1236–1265. URL: https://doi.org/10.1214/09-AOAS312, doi:10.1214/09-AOAS312. 2, 3

[vdMH08] VAN DER MAATEN L., HINTON G.: Visualizing data using t-sne. *Journal of machine learning research 9*, Nov (2008), 2579–2605. 1, 2

[VLL*10] VINCENT P., LAROCHELLE H., LAJOIE I., BENGIO Y., MANZAGOL P.-A.: Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research 11*, Dec (2010), 3371–3408. 2

[VWS14] VU N. T., WEINER J., SCHULTZ T.: Investigating the learning effect of multilingual bottle-neck features for ASR. In *Proc. Interspeech* (2014). 1

[WBR*16] WEBER P., BAI L., RUSSELL M. J., JANCOVIC P., HOUGHTON S. M.: Interpretation of low dimensional neural network bottleneck features in terms of human perception and production. In *Proc. Interspeech* (2016). 1

[YP19] YU S., PRÍNCIPE J. C.: Understanding autoencoders with information theoretic concepts. *Neural Networks 117* (2019), 104 – 123. doi:https://doi.org/10.1016/j.neunet.2019.05.003. 2