INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Distilling knowledge from Gaussian process teacher to neural network student

Jeremy H. M. Wong, Huayun Zhang, and Nancy F. Chen

Institute for Infocomm Research (I^{2}R), A*STAR, Singapore
{jeremy_wong,zhang_huayun,nfychen}@i2r.a-star.edu.sg

Abstract

Neural Networks (NN) and Gaussian Processes (GP) are different modelling approaches. The former stores characteristics of the training data in its many parameters, and then performs inference by parsing inputs through these parameters. The latter instead performs inference by computing a similarity between the test and training inputs, and then predicts test outputs that are correlated with the reference training outputs of similar inputs. These models may be combined to leverage upon their diversity. However, both combination and the matrix computations for GP inference are expensive. This paper investigates whether a NN student is able to effectively learn from the information distilled from a GP or ensemble teacher. It is computationally cheaper to infer using this student. Experiments on the speechocean762 spoken language assessment dataset suggest that learning is effective.

Index Terms: Gaussian process, neural network, knowledge distillation, ensemble combination, spoken language assessment

1. Introduction

Knowledge distillation [1, 2], also known as teacher-student learning, aims to train a student model to adopt the behaviour of one or more teacher models. This is useful for model compression [1], domain adaptation [3], and semi-supervised learning [1]. It can even compress an ensemble of multiple diverse models into a single compact model [2], to achieve accurate predictions at low computational cost.

The combined ensemble performance leverages upon the diversity of behaviour of the constituent models. Having different instances in which each ensemble member makes a mistake presents opportunities for the correct predictions of a model to amend for the errors of other models. One approach to encourage diversity is to design differences into the model structures. Such differences include using different parameter values [4] and Neural Network (NN) topologies [5]. Perhaps, even more diversity may arise between different types of models. This paper investigates combining the predictions of a NN with those of a Gaussian Process (GP) [6]. However, using multiple models may entail a high run-time computational cost. Furthermore, inference through a GP alone may be expensive, with a computational cost that may scale as the cube of the training set size. Through knowledge distillation, a student NN may be able to learn from the behaviours of this diverse ensemble, while having a cheaper run-time computational cost. Previous knowledge distillation works often investigate propagating information only between NN teachers and students [1, 2, 3, 7, 8, 9, 10]. This paper investigates the possibility of distilling knowledge from a GP teacher to a NN student. The proposal is evaluated on a Spoken Language Assessment (SLA) task.

The novelties proposed in this paper are:

- To first investigate the combination of a GP with a NN. This is needed to construct a teacher ensemble.
- Then more importantly, to assess the feasibility of distilling knowledge from a GP teacher or ensemble into a NN student.

This work was supported by the A*STAR Computational Resource Centre through the use of its high performance computing facilities.

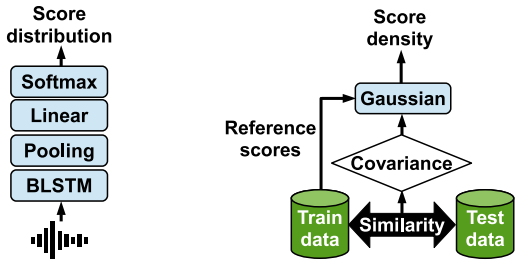
2. Related work

A GP and a NN are different model types to distil knowledge across, building upon prior works that seek to expand the diversity of models that can be distilled between. A large NN teacher can be compressed into a student with fewer layers and nodes [1]. A recurrent NN can be distilled into a feed-forward NN [7]. A Gaussian mixture model student can learn from a NN teacher [11]. Knowledge can be distilled between models that use different clusterings of context-dependent states [8]. Offline models can be distilled into causal models for streaming applications [9]. Knowledge can also be distilled between autoregressive and non-autoregressive speech recognition models [10]. In SLA, work in [12, 13] aims to improve tonal mispronunciation detection, by training a Mandarin tonal acoustic model student toward the soft tone posteriors generated by a teacher acoustic model, to address the inaccuracy of labelling non-native tones as one-hot targets.

This paper uses a combination of neural and non-neural models in two ways. First, an ensemble comprising a GP and a NN is combined over the output posteriors. Second, a NN is used to extract embeddings, which are then used as inputs to a GP, similarly to [14]. In the experiments here, the NN extractor is not fine-tuned jointly together with the GP, to avoid overfitting issues [15]. Other forms of neural and non-neural combinations are used in related fields. In speech recognition, NNs can be used to either extract features for a hidden Markov model with Gaussian mixture model emission likelihoods in a tandem model [16, 17], or to directly compute the hidden Markov model emission likelihoods in a hybrid model [18]. In diarisation, a NN may not generalise well to more speakers than were seen during training. This can be addressed by only using the NN to diarise short segments, and then using unsupervised clustering approaches to diarise across the segments [19]. A NN can also be used to extract speaker discriminative features, such as x-vectors [20], which can then be diarised using unsupervised clustering [21].

GPs can be computationally expensive to use at run-time, because of the need to perform multiplication and inversion on kernel matrices with dimensions equal to the training set size. This presents a barrier to using a GP with large diverse train-

10.21437/Interspeech.2023-190



(a) *Neural network* (b) *Gaussian process*

Figure 1: *Models for spoken language assessment*

ing datasets. The kernel dimension can be reduced by sparsifying the training data, to only consider a smaller set of inducing points [22]. The cost of matrix operations on the kernel can also be alleviated by reducing its rank, then training this low-rank GP student to approximate the full-rank GP teacher [23].

3. Gaussian process

A GP, depicted in figure 1b, computes training set outputs, $\boldsymbol{y}$, from training set inputs, $\mathbf{X}$, by assuming that there are latent variables, $\boldsymbol{f}$, that are Gaussian distributed with a covariance computed from the similarity between inputs,

$$p\left(\boldsymbol{f}|\mathbf{X}\right) = \mathcal{N}\left(\boldsymbol{f}; \mathbf{0}, \mathbf{K}\left(\mathbf{X}, \mathbf{X}\right)\right). \tag{1}$$

The outputs are assumed to be conditionally independent of the inputs when given these latent variables,

$$p\left(\boldsymbol{y}|\mathbf{X}\right) \approx \int p\left(\boldsymbol{y}|\boldsymbol{f}\right) p\left(\boldsymbol{f}|\mathbf{X}\right) d\boldsymbol{f}. \tag{2}$$

The kernel that represents the covariance in (1) measures a similarity between inputs, implying that nearby inputs should have latent variables that are correlated. This paper measures similarity using the squared exponential kernel,

$$k_{ij}\left(\mathbf{X}, \mathbf{X}\right) = s^2 \exp\left[-\frac{\left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)^\top \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)}{2l^2}\right], \tag{3}$$

where s and l are the scale and length hyper-parameters respectively, and i and j are data point indexes. The output can be chosen to be Gaussian distributed around the latent variable,

$$p\left(\boldsymbol{y}|\boldsymbol{f}\right) = \mathcal{N}\left(\boldsymbol{y}; \boldsymbol{f}, \sigma^2\mathbf{I}\right), \tag{4}$$

where σ is a hyper-parameter representing observation noise and $\mathbf{I}$ is the identity matrix. Substituting this into (2) equates to

$$p\left(\boldsymbol{y}|\mathbf{X}\right) = \mathcal{N}\left(\boldsymbol{y}; \mathbf{0}, \mathbf{K}\left(\mathbf{X}, \mathbf{X}\right) + \sigma^2\mathbf{I}\right), \tag{5}$$

which can be maximised over the training set to tune the hyper-parameters. In the marginal likelihood of (5), the mean of the output is independent of the input, because the chosen form of the GP prior in (1) also has a mean that is independent of the input. Instead, the form of the marginal likelihood is such that if two inputs are close together as measured by the kernel, then their outputs should be correlated. Thus, this marginal likelihood may not explicitly compute a mapping from the input to the output, in the way that a NN does, illustrated in figure 1a. What the GP hyper-parameters learn by maximising this marginal likelihood on the training data may be different from the input to output mapping learned by the NN parameters.

During run-time, the density of the hypothesised output, $\widehat{\boldsymbol{y}}$, can be computed from the test set inputs, $\widehat{\mathbf{X}}$, by first computing a density of the test set latent variables, $\widehat{\boldsymbol{f}}$, as

$$p\left(\widehat{\boldsymbol{f}}\middle|\widehat{\mathbf{X}}, \boldsymbol{y}, \mathbf{X}\right) = \frac{p\left(\boldsymbol{y}, \widehat{\boldsymbol{f}}\middle|\mathbf{X}, \widehat{\mathbf{X}}\right)}{p\left(\boldsymbol{y}|\mathbf{X}\right)} \tag{6}$$

$$= \mathcal{N}\left(\widehat{\boldsymbol{f}}; \widehat{\boldsymbol{\mu}}, \widehat{\mathbf{V}}\right), \tag{7}$$

where

$$\widehat{\boldsymbol{\mu}} = \mathbf{K}^{\hat{x}x}\left[\mathbf{K}^{xx} + \sigma^2\mathbf{I}\right]^{-1}\boldsymbol{y} \tag{8}$$

$$\widehat{\mathbf{V}} = \mathbf{K}^{\hat{x}\hat{x}} - \mathbf{K}^{\hat{x}x}\left[\mathbf{K}^{xx} + \sigma^2\mathbf{I}\right]^{-1}\mathbf{K}^{x\hat{x}}, \tag{9}$$

and

$$p\left(\boldsymbol{y}, \widehat{\boldsymbol{f}}\middle|\mathbf{X}, \widehat{\mathbf{X}}\right) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{y} \\ \widehat{\boldsymbol{f}}\end{bmatrix}; \mathbf{0}, \begin{bmatrix}\mathbf{K}^{xx} + \sigma^2\mathbf{I} & \mathbf{K}^{x\hat{x}} \\ \mathbf{K}^{\hat{x}x} & \mathbf{K}^{\hat{x}\hat{x}}\end{bmatrix}\right). \tag{10}$$

The notations of $\mathbf{K}^{xx} = \mathbf{K}(\mathbf{X}, \mathbf{X})$, $\mathbf{K}^{\hat{x}\hat{x}} = \mathbf{K}(\widehat{\mathbf{X}}, \widehat{\mathbf{X}})$, and $\mathbf{K}^{x\hat{x}} = \mathbf{K}(\mathbf{X}, \widehat{\mathbf{X}}) = \mathbf{K}^{\hat{x}x\top}$ are used for brevity. The output density function is computed as

$$p\left(\widehat{\boldsymbol{y}}\middle|\widehat{\mathbf{X}}, \boldsymbol{y}, \mathbf{X}\right) = \int p\left(\widehat{\boldsymbol{y}}\middle|\widehat{\boldsymbol{f}}\right) p\left(\widehat{\boldsymbol{f}}\middle|\widehat{\mathbf{X}}, \boldsymbol{y}, \mathbf{X}\right) d\widehat{\boldsymbol{f}} \tag{11}$$

$$= \mathcal{N}\left(\widehat{\boldsymbol{y}}; \widehat{\boldsymbol{\mu}}, \widehat{\mathbf{V}} + \sigma^2\mathbf{I}\right). \tag{12}$$

The hypothesised output can then be inferred by computing the mean, mode, or median of (12), which are equivalent, because of the symmetry and unimodality of a Gaussian.

4. Spoken language assessment setup

The proposed approach is evaluated on a Spoken Language Assessment (SLA) setup. The motivations for this choice are as follows. First, SLA is a regression task, but is often treated in a somewhat classification manner through the rounding of the reference and hypothesis scores. This allows a reasonable match between the use of a categorical NN and a standard GP. Second, the training set size for SLA is often within a range that is manageable for a GP, during both hyper-parameter optimisation and run-time inference, without needing to resort to sparse approximations. This alleviates any impact that such sparse approximations may have on the experimental trends. Third, previous work in [24] has already investigated applying a GP to SLA, demonstrating a proof of concept.

SLA aims to predict a score for an input spoken audio that is related to the oral proficiency of the speaker. Aspects of oral proficiency that are often assessed include pronunciation accuracy, intonation, fluency, prosody, sentence completeness, task completion, and topic relevance. A model in automatic SLA is trained to compute a score that is similar to one that an expert human rater would have assigned.

The experiments used the speechocean762 dataset [25]. The training and test sets each encompass 2500 sentences, spoken by 125 disjoint speakers. The speakers read the sentences in English, but are natively from a Mandarin speaking background. The data is annotated with proficiency scores at the phoneme, word, and sentence levels, but only the sentence scores were used in the experiments here. Each sentence is annotated by 5 human raters on a range from 0 to 10, assessing the pronunciation accuracy, fluency, prosody, and sentence completeness. The experiments were replicated across the pronunciation accuracy, fluency, and prosody scores to validate that

the trends generalise across score types. The sentence completeness was avoided, because of the label imbalance. A single scalar reference for each sentence was computed as the average of the scores from the 5 raters. This differs from [25], which instead computes the median. As is done in [25], the hypothesis and reference scores were first rounded to the closest integers, before computing the evaluation metrics. The model's performance was assessed using the Pearson's Correlation Coefficient (PCC) and Mean Squared Error (MSE), following [25]. A two-tailed paired t-test was used to assess the statistical significance between MSE values. It is non-trivial to use a t-test to compare PCCs, because it is not expressed as a sum over data points, and thus eludes application of the central limit theorem. Instead, an approximately normally distributed transformation was first computed from the pair of PCCs being compared [26], then the two-tailed cumulative density was reported as the significance. This approach is referred to as Z_1^* in [27].

To perform SLA on read speech, the audio was first forced aligned to the reference transcript, using a hybrid model that was trained on the 960 hours Librispeech data [28], following [25]. This forced alignment was used to compute Goodness Of Pronunciation (GOP) [29], Log Phone Posterior (LPP) [30], Log Posterior Ratio (LPR) [30], and tempo [31] features for each phoneme. For each phoneme, pitch features [32] were extracted per frame, then averaged over all frames in the phoneme. Phoneme embeddings [31] were extracted from a 32-node recurrent NN continuous skip-gram model [33], trained on the non-silence phoneme sequences. Following [31], these features were concatenated to form one vector per phoneme in the sentence.

NN SLA models used a sequence of these feature vectors as input to a Bidirectional Long Short-Term Memory (BLSTM) layer, with 32 nodes per direction. Equal-weighted mean pooling was used to compute a sentence-level embedding from the BLSTM output sequence. A linear layer then computed the desired output dimension from this embedding. Either a 1-dimensional linear output was parsed through a sigmoid and scaled to the score range to be trained with MSE as a regression model, or an 11-dimensional linear output was parsed through a softmax and trained with Cross-Entropy (CE) for a categorical classification model. A dropout [34] rate of 60% was used at the inputs to the BLSTM and linear layers. A 10% held out validation set was used for NN training, without speaker disjointment. During run-time, a scalar hypothesis score was inferred from the categorical NN as the mean of the discrete output posterior.

A GP does not naturally work with sequential inputs. The kernel can be modified to compute similarities over sequences [35] or hand-crafted sentence-level features can be extracted [24]. A NN can also be used to extract features as inputs to a GP [14]. In this paper, the sentence-level embedding from the pooling layer of the NN was used as the input to the GP. Embeddings from the MSE-trained regression NN were used, as initial tests suggested that these yielded slightly better GP performance than when using embeddings from the categorical NN, though not significantly. The GP hyper-parameters were optimised by maximising the marginal likelihood of (2), using gradient descent.

5. Ensemble combination

A NN stores learned knowledge about the training data into its parameters, and then leverages upon this learned input to output mapping during run-time. This requires there to be a sufficient quantity of parameters to capture the complexities of the training data. As opposed to this, the GP only has three hyper-parameters that can be tuned toward the data. These may not suffice to learn much about the training data. Instead, the GP uses the training data itself during run-time, by comparing the similarities between the test set and training set inputs, to hypothesise test set outputs that correlate with the reference outputs of nearby training set inputs. These different approaches to data processing may yield diverse error patterns in their hypotheses.

The categorical-output NN computes discrete posteriors over the output classes, which for SLA represent the integer scores. The GP here is designed for regression, and thus computes a likelihood over continuous scores. This reveals another difference between the chosen NN and GP designs, in that the categorical NN assumes neither monotonicity nor linearity of the output space, while the GP does. For consistency, the GP's output density of (12) is first discretised to look more like a classification model, by cumulating over the continuous scores that would have been rounded to each integer,

$$P_{\text{GP}}\left(c|\widehat{\boldsymbol{x}}_i\right) = \frac{\int_{c-0.5}^{c+0.5} p\left(\widehat{y}_i|\widehat{\boldsymbol{x}}_i, \boldsymbol{y}, \mathbf{X}\right) d\widehat{y}_i}{\sum_{c'} \int_{c'-0.5}^{c'+0.5} p\left(\widehat{y}'_i|\widehat{\boldsymbol{x}}_i, \boldsymbol{y}, \mathbf{X}\right) d\widehat{y}'_i}, \qquad (13)$$

where c are the possible integer scores. Unlike a categorical NN, this discrete posterior assumes monotonicity of the scores. This method of discretisation matches the practice of rounding the hypothesised scores before evaluation [25]. In (13), equal bin widths are used, as opposed to allowing the extreme bins to span supports up till $\pm\infty$, to avoid biasing the score toward these extremes. As a result, integration is not performed over the infinite support of the Gaussian, and thus the denominator in (13) is needed for $P_{\text{GP}}\left(c|\widehat{\boldsymbol{x}}_i\right)$ to sum to one. An initial test comparing inference from the GP as either a discrete mean of (13) or a continuous mean of (12) did not yield any significant performance difference. It is possible to formulate the GP to perform multi-class classification [6] directly, but this is more computationally expensive, requires approximation in its implementation, and may forego the monotonicity assumption.

This paper considers a combination over probability distributions. The discrete posteriors from the GP and NN are combined as a mixture,

$$\overline{P}\left(c|\widehat{\boldsymbol{x}}_i\right) = \lambda P_{\text{GP}}\left(c|\widehat{\boldsymbol{x}}_i\right) + (1-\lambda)\, P_{\text{NN}}\left(c|\widehat{\boldsymbol{x}}_i\right), \qquad (14)$$

where $P_{\text{NN}}\left(c|\widehat{\boldsymbol{x}}_i\right)$ is the NN posterior, and λ is the combination weight, such that $0 \leq \lambda \leq 1$. The combined hypothesis can then be inferred from the combined posterior, $\overline{P}\left(c|\widehat{\boldsymbol{x}}_i\right)$. In this paper, the hypothesis was inferred as the mean of the posterior, as initial tests suggested a better performance for SLA than inferring as the mode.

Combination over the outputs of diverse models may be viewed as a Monte Carlo approximation of a Bayesian NN [36]. Such combination aims to reduce the prevalence of model uncertainty, exemplified by the diversity of predictions due to the biases of different models, and yield a purer representation of data uncertainty in the combined output. A GP on its own may also be interpreted as a marginalisation over a distribution of functions [6]. However, all functions represented within the GP are influenced by the same choices made about the GP's design, and thus may be similarly biased. It may therefore still be beneficial to combine a GP with a different model type.

The combination results are shown in table 1. A combination weight of $\lambda = 0.5$ was used, thereby assuming equal importance between the GP and NN. The results suggest that

Table 1: *Combination of a GP and a NN*

Score type	Model	PCC↑	MSE↓
accuracy	NN	0.701	1.208
	GP	0.710	1.149
	NN + GP	0.724	1.085
fluency	NN	0.754	0.810
	GP	0.779	0.727
	NN + GP	0.784	0.706
prosody	NN	0.753	0.798
	GP	0.775	0.722
	NN + GP	0.791	0.669

the GP performs comparably against the NN. The combined ensemble outperforms both the NN and GP. The significances of the performance differences between the GP and the combination are $\rho_{\text{PCC}} = 0.005$ and $\rho_{\text{MSE}} = 0.002$ for pronunciation accuracy, $\rho_{\text{PCC}} = 0.291$ and $\rho_{\text{MSE}} = 0.168$ for fluency, and $\rho_{\text{PCC}} < 0.001$ and $\rho_{\text{MSE}} < 0.001$ for prosody. The combination gains may be significant for pronunciation accuracy and prosody. This suggests that the behaviours of a NN and a GP can be complementary.

6. Knowledge distillation

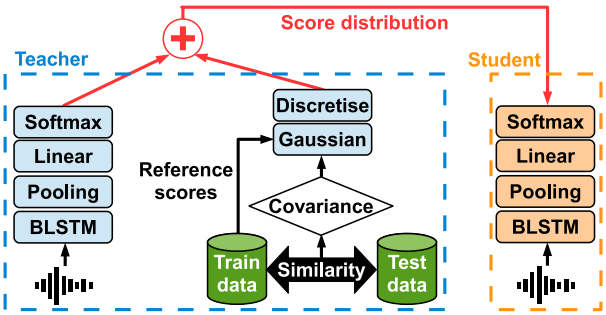


Figure 2: *Knowledge distillation from a teacher ensemble containing a Gaussian process, to a student neural network*

Processing multiple models in an ensemble can be computationally expensive. Furthermore, the GP itself may incur a high run-time cost. The GP requires training data to be processed while performing inference. This may entail a high computational cost that scales quadratically with the training set size for the matrix multiplications in (8) and (12). The matrix inverse in these equations, which can be pre-computed before run-time, has a computational cost that scales cubically with the training set size. Sparse approximations have previously been investigated to reduce the number of required training data points to be used at run-time and reduce this computational cost [22]. The run-time cost of a NN does not depend on the training set size. Distilling knowledge from a GP or ensemble into a single NN may thus capture the GP or ensemble behaviour, without requiring the expensive run-time computation. This is shown in figure 2. To perform knowledge distillation, a categorical NN student model is trained by minimising the discrete Kullback-Leibler (KL) divergence between the teacher's and student's posteriors [1],

$$\mathcal{F}_{\text{KL}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c} P_{\text{teacher}}\left(c|\boldsymbol{x}_i\right)\log P_{\text{student}}\left(c|\boldsymbol{x}_i\right), \quad (15)$$

Table 2: *NN student performance when trained toward a single NN or GP teacher, or toward an ensemble of both*

Score type	Teacher	NN student performance PCC	MSE
accuracy	NN	0.701	1.208
	GP	0.719	1.119
	NN + GP	0.728	1.089
fluency	NN	0.757	0.807
	GP	0.794	0.680
	NN + GP	0.791	0.692
prosody	NN	0.755	0.803
	GP	0.800	0.659
	NN + GP	0.796	0.666

where N is the training set size.

Table 2 shows the student performance when training a single student NN toward either a single NN or GP teacher, or toward an ensemble combination of both using (14). The results suggest that a NN student is able to learn effectively from a GP teacher, with the student surpassing the teacher's performance on both metrics in this instance, though not always significantly, with $\rho_{\text{PCC}} = 0.150$ and $\rho_{\text{MSE}} = 0.316$ for pronunciation accuracy, $\rho_{\text{PCC}} = 0.007$ and $\rho_{\text{MSE}} = 0.051$ for fluency, and $\rho_{\text{PCC}} < 0.001$ and $\rho_{\text{MSE}} = 0.006$ for prosody. This demonstrates that knowledge distillation is able to propagate information over the different model types. The single NN student is also able to closely match the performance of an ensemble of NN and GP teachers. Thus, knowledge distillation can be used to yield a level of performance close to that of the ensemble, while only requiring the run-time cost of a single NN. Initial tests did not yield any improvements when training the student with an interpolation of the CE criterion.

7. Conclusion

This paper has explored knowledge distillation from a GP teacher or ensemble of NN and GP teachers, to a NN student. The experiments suggest that a NN student is able to learn effectively from a GP teacher. The NN student can thus be used during run-time, thereby requiring a cheaper computation cost than a GP or ensemble.

A GP is formulated to express greater uncertainty for test inputs that reside far from the training inputs, represented in the variance of (9). This expression of distributional uncertainty [37] may seem reasonable. As opposed to this, a NN may express high confidence in its hypothesis for out-of-domain data [38]. A future investigation may consider whether a NN is able to learn this desirable distributional uncertainty behaviour from a GP. Perhaps, this may be approached by training a student NN toward a GP teacher on unlabelled data that is mismatched with the domain of the training data.

8. References

[1] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size DNN with output-distribution-based criteria," in *Interspeech*, Singapore, Sep 2014, pp. 1910–1914.

[2] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *Deep Learning and Representation Learning Workshop, NIPS*, Montréal, Canada, Dec 2014.

[3] J. Li, M. L. Seltzer, X. Wang, R. Zhao, and Y. Gong, "Large-scale domain adaptation via teacher-student learning," in *Interspeech*, Stockholm, Sweden, Aug 2017, pp. 2386–2390.

[4] L. K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993–1001, Oct 1990.

[5] L. Deng and J. C. Platt, "Ensemble deep learning for speech recognition," in *Interspeech*, Singapore, Sep 2014, pp. 1915–1919.

[6] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[7] W. Chan, N. R. Ke, and I. Lane, "Transferring knowledge from a RNN to a DNN," in *Interspeech*, Dresden, Germany, Sep 2015, pp. 3264–3268.

[8] J. H. M. Wong and M. J. F. Gales, "Student-teacher training with diverse decision tree ensembles," in *Interspeech*, Stockholm, Sweden, Aug 2017, pp. 117–121.

[9] G. Kurata and G. Saon, "Knowledge distillation from offline to streaming RNN transducer for end-to-end speech recognition," in *Interspeech*, Shanghai, China, Oct 2020, pp. 2117–2121.

[10] X. Gong, Z. Zhou, and Y. Qian, "Knowledge transfer and distillation from autoregressive to non-autoregressive speech recognition," in *Interspeech*, Incheon, South Korea, Sep 2022, pp. 2618–2622.

[11] J. H. M. Wong, M. J. F. Gales, and Y. Wang, "Learning between different teacher and student models in ASR," in *ASRU*, Singapore, Dec 2019, pp. 93–99.

[12] W. Li, N. F. Chen, S. M. Siniscalchi, and C.-H. Lee, "Improving Mandarin tone mispronunciation detection from non-native learners with soft-target tone labels and BLSTM-based deep models," in *ICASSP*, Calgary, Canada, Apr 2018, pp. 6249–6253.

[13] ——, "Improving mispronunciation detection of Mandarin tones for non-native learners with soft-target tone labels and BLSTM-based deep tone models," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2012–2024, Dec 2019.

[14] R. Salakhutdinov and G. E. Hinton, "Using deep belief nets to learn covariance kernels for Gaussian processes," in *NIPS*, Vancouver, Canada, Dec 2007.

[15] S. W. Ober, C. E. Rasmussen, and M. van der Wilk, "The promises and pitfalls of deep kernel learning," in *UAI*, Jul 2021, pp. 1206–1216.

[16] S. Sharma, D. P. W. Ellis, S. Kajarekar, P. Jain, and H. Hermansky, "Feature extraction using non-linear transformation for robust speech recognition on the AURORA database," in *ICASSP*, Istanbul, Turkey, Jun 2000, pp. 1117–1120.

[17] D. P. W. Ellis, R. Singh, and S. Sivadas, "Tandem acoustic modeling in large-vocabulary recognition," in *ICASSP*, Salt Lake City, USA, May 2001, pp. 517–520.

[18] H. A. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Kluwer Academic Publishers, 1994.

[19] S. Horiguchi, S. Watanabe, P. García, Y. Xue, Y. Takashima, and Y. Kawaguchi, "Towards neural diarization for unlimited numbers of speakers using global and local attractors," in *ASRU*, Cartagena, Colombia, Dec 2021, pp. 98–105.

[20] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: robust DNN embeddings for speaker recognition," in *ICASSP*, Calgary, Canada, Apr 2018, pp. 5329–5333.

[21] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *ICASSP*, New Orleans, USA, Mar 2017, pp. 4930–4934.

[22] M. Bauer, M. van der Wilk, and C. E. Rasmussen, "Understanding probabilistic sparse Gaussian process approximations," in *NIPS*, Barcelona, Spain, Dec 2016.

[23] C. Song and Y. Sun, "Kernel distillation for fast Gaussian processes prediction," in *BNP@NeurIPS*, Montréal, Canada, Dec 2018.

[24] R. C. van Dalen, K. M. Knill, and M. J. F. Gales, "Automatically grading learners' English using a Gaussian process," in *SLaTE*, Leipzig, Germany, Sep 2015, pp. 7–12.

[25] J. Zhang, Z. Zhang, Y. Wang, Z. Yan, Q. Song, Y. Huang, K. Li, D. Povey, and Y. Wang, "speechocean762: an open-source non-native English speech corpus for pronunciation assessment," in *Interspeech*, Brno, Czechia, Aug 2021, pp. 3710–3714.

[26] O. J. Dunn and V. Clark, "Correlation coefficients measured on the same individuals," *Journal of the American Statistical Association*, vol. 64, no. 325, pp. 366–377, Mar 1969.

[27] J. H. Steiger, "Tests for comparing elements of a correlation matrix," *Psychological Bulletin*, vol. 87, no. 2, pp. 245–251, 1980.

[28] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *ICASSP*, Brisbane, Australia, Apr 2015, pp. 5206–5210.

[29] S. M. Witt and S. J. Young, "Phone-level pronunciation scoring and assessment for interactive language learning," *Speech Communication*, vol. 30, no. 2-3, pp. 95–108, Feb 2000.

[30] W. Hu, Y. Qian, F. K. Soong, and Y. Wang, "Improved mispronunciation detection with deep neural network trained acoustic models and transfer learning based logistic regression classifiers," *Speech Communication*, vol. 67, pp. 154–166, Mar 2015.

[31] H. Zhang, K. Shi, and N. F. Chen, "Multilingual speech evaluation: case studies on English, Malay and Tamil," in *Interspeech*, Brno, Czechia, Aug 2021, pp. 4443–4447.

[32] P. Ghahremani, B. BabaAli, D. Povey, K. Riedhammer, J. Trmal, and S. Khudanpur, "A pitch extraction algorithm tuned for automatic speech recognition," in *ICASSP*, Florence, Italy, May 2014, pp. 2494–2498.

[33] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *ICLR*, Scottsdale, USA, May 2013.

[34] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, Jun 2014.

[35] H. Lodhi, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," in *NIPS*, Denver, USA, Nov 2000, pp. 563–569.

[36] D. J. C. MacKay, "Bayesian interpolation," *Neural Computation*, vol. 4, no. 3, pp. 415–447, May 1992.

[37] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," in *NeurIPS*, Montréal, Canada, Dec 2018.

[38] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *ICML*, Sydney, Australia, Aug 2017, pp. 1321–1330.