# Using Deep Bidirectional Recurrent Neural Networks for Prosodic-Target Prediction in a Unit-Selection Text-to-Speech System

*Raul Fernandez*[1], *Asaf Rendel*[2], *Bhuvana Ramabhadran*[1], *Ron Hoory*[2]

[1]TJ Watson Research Center, Yorktown Heights, NY – USA
[2]IBM Haifa Reseach Lab, Haifa – Israel

fernanra@us.ibm.com, asafren@il.ibm.com

10.21437/Interspeech.2015-372

## Abstract

Deeply-stacked Bidirectional Recurrent Neural Networks (BiRNNs) are able to capture complex, short- and long-term, context dependencies between predictors and targets due to the non-linear dependency they introduce on the entire observation when predicting a target, thanks to the use of recurrent hidden layers that accumulate information from all preceding and future observations. This aspect of the model makes them desirable for tasks such as the prediction of prosodic contours for text-to-speech systems, where the surface prosody can be a result of the interaction between local and non-local features. Although previous work has demonstrated that they attain state-of-the-art performance for this task within a parametric synthesis framework, their use within unit-selection synthesis systems remains unexplored. In this work we deploy this class of models within a unit selection system, investigate their effect on the outcome of the unit search, and perceptually evaluate it against the baseline (decision-tree-based) approach.

**Index Terms**: speech synthesis, unit selection, recurrent neural networks, deep learning

## 1. Introduction

Unit-selection, concatenative text-to-speech (TTS) systems have the ability to reproduce to some extent the natural prosody inherent in the segments, and for this reason can generate synthesis that is perceived as more natural than that generated by parametric systems relying exclusively on a statistical model. This naturalness, on the other hand, is often fairly "local" and suffers from concatenation artifacts when the search component retrieves segments that, once joined together, result in awkward prosodic turns which can persist even when signal processing is used to smooth the transition across them. Most state-of-the-art concatenative systems employ some underlying statistical prosody-generation model. Although they can be used differently within different architectures, there are at least two ways in which the target prosody predicted by these models can be used in a unit-selection system. First, a unit-search objective function can explicitly attempt to match prosodic properties (usually through an additive prosodic component, properly weighted) and partially penalize units in the inventory according to how much they deviate from the predicted targets. If the architecture implements a post-search algorithm that favors the natural prosody of the retrieved units, then the predicted prosodic contours function only as a proxy to suitably bias the search toward them. In these types of architectures, they could, at this point, be discarded. The second approach, on the other hand, is to retain these contours after the search is carried out, and process the units to prosodically match the predicted targets. Such an approach almost invariably introduces processing artifacts that can degrade the naturalness of the speech. This operating condition, however, of interest to us for the following reason: a limited inventory of units cannot reproduce all the desirable combinations of acoustic units under various prosodic contexts, and being subject to the existing prosody in the corpus can severely limit the expressive range of the engine. Consider that a state-of-the-art TTS system designed to synthesize speech in a fairly *neutral* style typically uses a corpus of 10-15 hours of speech, or more. Expecting synthesis outside of this neutral range would require enlarging the base corpus to reflect the variability of interest and populate the search space. This is not always feasible or even possible.

Motivated by the preceding observations, we are interested in investigating the effects of a state-of-the-art prosody-prediction model with a unit-selection system. We will adopt a class of dynamic models that has been recently receiving attention for prosody-prediction in TTS systems: deeply-stacked Bidirectional Recurrent Neural Networks (BiRNNs) [1, 2]. These models are able to capture complex, short- and long-term, context dependencies between predictors and targets due to the non-linear dependency they introduce on the entire observation when predicting a target, thanks to the use of recurrent hidden layers that accumulate information from all preceding and future observations. This makes them suitable for tasks such as the prediction of prosodic contours, where the surface prosody can be a result of the interaction between local and non-local features. As a baseline model we will adopt Random Decision Tree Forests, which, due to the boosted performance of ensemble averaging, is representative of a strong baseline, but also reflects the decision-tree methodology underlying many prosody-prediction models in TTS.

To address our investigation, we will, in particular, look at the following questions:(Q1) allowing for signal modification, can BiRNN models outperform baseline models in a perceptual evaluation; (Q2) what is the effect on the output of the unit search of this improved prosodic-target contour; and (Q3) how do the baseline and BiRNN approaches compare when prosodic targets are used to drive the unit-selection search and then the natural prosody of the units is favored in the output waveform.

The paper is structured as follows. We begin by reviewing previous related work in this area in Section 2. After providing an overview of the underlying system's architecture in 3, and the corpus used for this work in Section 4, we present the two modeling approaches in Section 5. These two approaches are compared and evaluated in 6 before concluding in Section 7.

## 2. Previous and Related Work

The authors investigated (Q1) in the context of parametric systems where spectral sequences were identical [1], and showed improvements for BiRNN-generated contours when the output synthesis differed only in prosody. The issue within unit-selection systems, however, is different, as we have tried to argue, since the prosodic targets stand to both influence the sequence of acoustic units selected, as well as the final prosody. The remaining two questions (Q2-3) are motivated by the current unit-selection scenario, and remain, to the best of our knowledge, unexplored in the literature. The closest work to ours is that of Fan et al. [2], who also investigated using BiRNNs for speech synthesis, and Zen and Sak [3], who look at a unidirectional version that allows for streaming. In both of these works, however, the focus is on a parametric framework. Parametric synthesis, in fact, is where most of the recent applications of (deep) neural networks have taken place, such as in related work by Zen et al. [4], looking at non-dynamic versions for estimating unimodal parametric targets, and by Zen and Senior [5] for estimating the parameters in the mixture-density case.

## 3. Architecture

The underlying speech synthesis engine used in this work to investigate the effect of the two prosody-prediction approaches is the IBM concatenative unit-selection system. Due to space constraints, we will only review aspects of this architecture that are most directly relevant to this work, and refer the reader to [6] for a general system description.

The unit selection in this system is carried out by a Viterbi search over a trellis, where each time slice is populated with state-sized acoustic units returned by an acoustic model. (The units correspond to roughly 1/3 of a phone and result from forced-alignments with 3-state hidden Markov models). To traverse this trellis and extract the optimal sequence, an additive cost function is defined to include (i) a prosodic target cost $C_{ta}$, and (ii) a concatenation cost $C_{co}$. Whereas $C_{co}$ is a function of the spectral and pitch differences between any two adjacent candidates and can be computed directly from the tokens, the target cost $C_{ta}$ evaluates how the natural prosodic properties of a unit deviate from those predicted by a model, and consists of a weighted sum of the following 3 subcomponents (where each of weights can be specified or tuned):

- $f_0$ cost: it is the absolute difference between the $f_0$ of the target and that of the token, divided by the smaller of the two; this cost is assessed at the start and end of the unit and added up.
- duration cost: absolute difference between the duration of the target and that of the token, divided by that of the token
- energy cost: it is zero, except when the target energy is much larger than that of the token (this component is usually negligible compared to the $f_0$ and duration costs)

From this description of how the search is carried out, we see that the task of a prosody model within this architecture is to generate four targets for each acoustic unit: duration, energy, plus initial and final $f_0$.

After the search retrieves a sequence of units, the pitch-synchronous overlap-add (PSOLA) algorithm is used to concatenate the units together subject to some prosodic specifications. Two variants of the prosody handed out to the waveform-generation module are discussed in this paper:

1. A pitch contour is specified by linearly interpolating the target $f_0$ values and then smoothing the piecewise linear function with a decaying exponential kernel $h(\tau) = \frac{1}{2\tau_0} \exp -|\tau/\tau_0|$, where $\tau_0$ is a time constant set to $0.03$ in this implementation. The target duration and target energy are used.

2. When the output of the search contains any sequence of more than 5 units which occurred contiguously in the database, this contiguous region (CR) is not subject to *any* prosodic modification, except at the boundaries where the CR meets other units. An interpolation is carried out to seamlessly blend the natural $f_0$ from within the CR with that from outside, across 2 units on either side of the boundary. Outside any CRs, the $f_0$ consists of the natural $f_0$ track of the selected units, smoothed as per the procedure described above to eliminate jumps. Notice that in this mode, $f_0$ modification only takes place over (i) regions that are not contiguous (where $f_0$ is modified to the smoothed natural pitch) , and (ii) the first/last 2 units after/before a CR boundary. In this variant of the algorithm, all target durations are discarded in favor of the natural duration of the units (irrespective of whether they appear in a CR). The energy is modified to the predicted target energy over any non-CRs, as well as over the first/last 2 units after/before a CR boundary. We refer to the variant just described as the Bypass Algorithm.

The Bypass Algorithm allows mostly the natural prosody (including micro-prosodic effects present in the CRs) to be synthesized on the output, subject only to some modest amount of smoothing needed to blend the units. The first variant (which we will describe as the No Bypass condition), on the other hand, favors the target prosody, after some smoothing of the target $f_0$ contour.

## 4. Corpus

A concatenative speech database consisting of over 11 hours of professionally recorded speech from a native speaker of US English has been used as data to train prosodic models. The observations for these models correspond to state-sized units resulting from forced-alignment of the corpus using 3-state hidden Markov models. 90% of the utterances were used for training the models, and 10% as a validation set, resulting in sets of approximately 1.5M and 166K observations respectively. A feature set is built by extracting various categorical (e.g., part of speech, phrase type, etc.) and standard positional features (e.g., number of phones/words to/from a phrase/utterance boundary, etc.) commonly used for prosody prediction in TTS. Feature values are propagated down to the constituent states, and any categorical features are re-encoded using One-of-N binary vectors, resulting in a 336-dimensional input space. The four prosodic targets already introduced are also extracted for each observation, and paired with the inputs. For training the models, any numerical (non-Boolean) features, and all the targets, are z-scored with respect to the training mean and standard deviation. The prosodic targets generated by these models can then used at run time to synthesize test utterances that do not occur in the concatenative corpus, using either of the waveform-generation settings described in Section 3.

# 5. Prosody-Modeling Approaches

## 5.1. Bidirectional Recurrent Neural Networks with Long Short-Term Memory Units

An RNNs is a dynamic neural network that is in principle able to exploit structure in the input across various time lags by the use of recurrent hidden-unit layers that update their activations as a function of the previous or next hidden-state value, or, in the case of bidirectional RNNs, both. It is an alternative to models that use a finite amount of (usually local) input aggregation to address contextual effects, and is well-motivated for tasks such as prosody modeling, where short- and long-term properties of the input can affect the prosodic realization. Additionally, by stacking several such bidirectional layers, we are able to obtain models that are deep across time and layers, benefiting from the same compositionality of standard (static) deep neural networks.

Implementing such structures with standard hidden activation units (such as sigmoid or hyperbolic tangent), however, is usually insufficient to overcome training issues like the vanishing gradient, a problem that becomes exacerbated when BiRNNs are trained using algorithms like back-propagation through time (BPTT). One proposal to alleviate this well-known obstacle was the introduction of a compound memory cell called the Long Short-Term Memory (LSTM) unit [7, 8, 9], shown in Fig. 1. Such a cell keeps an internal unit, called the

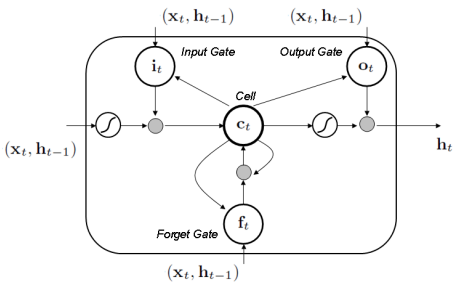


Figure 1: *LSTM gate with peephole connections, showing the internal structure and relation between the cell and the different gates. Solid nodes are multipliers. The mathematical operations carried out by this unit are detailed in Eqs. 1- 5*

"Constant Error Carousel" (CEC) which recycles activation and error signals for the input history, and is thus able to provide short-term memory storage for extended time lags. LSTM cells also contain 3 internal gates, known as the *input*, *forget* and *output* gates which output a value in $(0, 1)$ to modulate the input and help determine, respectively, (i) when an input is relevant enough to the task to be remembered, (ii) when an input is no longer relevant, and (iii) when the network should output a value. To date, most of the success of RNNs has been demonstrated when coupled with LSTM units [10]; this is the approach we follow in this work.

Mathematically, such gate implements the activation $\mathbf{h}_t = \phi_f(\mathbf{x}_t, \mathbf{h}_{t-1})$ through the following compound recursive equations [11]:

$$\mathbf{i}_t = \sigma(W_{xi}\mathbf{x}_t + W_{hi}\mathbf{h}_{t-1} + W_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \quad (1)$$

$$\mathbf{f}_t = \sigma(W_{xf}\mathbf{x}_t + W_{hf}\mathbf{h}_{t-1} + W_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f) \quad (2)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tau(W_{xc}\mathbf{x}_t + W_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (3)$$

$$\mathbf{o}_t = \sigma(W_{xo}\mathbf{x}_t + W_{ho}\mathbf{h}_{t-1} + W_{co}\mathbf{c}_t + \mathbf{b}_o) \quad (4)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tau(\mathbf{c}_t), \quad (5)$$

where $\mathbf{i}$, $\mathbf{f}$, $\mathbf{c}$ and $\mathbf{o}$ are the input gate, forget gate, cell gate, and output gate activation vectors respectively, all of the same size as the hidden vector $\mathbf{h}$; $W_{x\star}$ are the input-to-gate weights; $W_{h\star}$ are the hidden-to-hidden weights for the various gates; $W_{c\star}$ are diagonal matrices containing feedback connection weights from the cell to the gates (known as "peephole" weights); $\mathbf{b}_\star$ are the bias vectors; $\odot$ is the element-wise multiplication operator; $\sigma()$ is a function bound by $(0, 1)$ (sigmoid in this implementation) realizing a differentiable version of a "hard" gating function; and $\tau()$ is a differentiable non-linearity (hyperbolic tangent in this work).

Equations1-5 are defined for a cell on the forward chain; an analogously defined $\phi_b(\mathbf{x}_t, \mathbf{h}_{t+1})$ would operate on the backward chain. Deep bi-directional structures can be built by joining the forward and backward activations and feeding them as input to another recurrent, LSTM layer. If we let $\overrightarrow{\mathbf{h}}^n$ and $\overleftarrow{\mathbf{h}}^n$ be the activations of the forward and backward chains on the $n^{th}$ level of a stack, then

$$\overrightarrow{\mathbf{h}}_t^n = \phi_f(\mathbf{z}_t^n, \overrightarrow{\mathbf{h}}_{t-1}^n) \text{ and } \overleftarrow{\mathbf{h}}_t^n = \phi_b(\mathbf{z}_t^n, \overleftarrow{\mathbf{h}}_{t+1}^n), \quad (6)$$

where $\mathbf{z}_t^0 = \mathbf{x}_t$ and $\mathbf{z}_t^n = [\overrightarrow{\mathbf{h}}_t^{n-1}; \overleftarrow{\mathbf{h}}_t^{n-1}]$ for $n \geq 1$. The output on the final linear-output layer is then

$$\mathbf{y}_t = W_{\overrightarrow{h}y}\overrightarrow{\mathbf{h}}^N + W_{\overleftarrow{h}y}\overleftarrow{\mathbf{h}}^N + \mathbf{b}_y. \quad (7)$$

The Theano symbolic-expression library [12, 13] has been used in our implementation to define the operations in Eqns. 1-7, and to optimize the corresponding weights with respect to the square-loss function using stochastic gradient descent and BPTT. The network structure tested here consists of 3 stacks of bidirectional hidden layers, containing 67, 57, and 46 LSTM units per layer respectively.

## 5.2. Random Forests

We have adopted random-forest ensembles of regression trees [14] as a baseline model to predict the four prosodic targets. This choice is motivated by the fact that they represent a strong baseline, (due to the benefits that ensemble-model averaging provide in correcting for the high-variance of single regression-tree predictions) while at the same time containing the decision-tree machinery that underlies much of the prosodic-target prediction models in many TTS system.

A forest consist of an ensemble of $N$ independently built, random trees for regression, where each random tree is grown by considering a randomly-drawn subset of $p$ predictors (out of a maximum of $P$) at each node split. Once grown, an input feature is evaluated using the $N$ models in the ensemble, and the ensemble average used to generate a final prediction. In this implementation, a random forest is built for each of the targets, and all of the training data is used to build each tree. The hyper-parameters $p$ and $N$ are tuned using the validation set over the following range: $p$ is searched over 10 values in the range $[\lfloor 0.1P \rfloor, \cdots, P]$ at intervals of $\lfloor 0.1P \rfloor$; $N$ is searched over the integers in the range $[2, 10]$.

# 6. Model Comparison

A set of 40 sentences was used as a test for comparing the effect of the two prosodic models. These sentences are not contained in the synthesis database, and reflect content from various domains, such as news, promotional material, and computer-assisted prompts. The texts were synthesized using both the

BiRNN and Random Forest models as underlying prosodic-target generators, and under the two variants of the bypass algorithm (with and with no bypass). During the search, each model was used to generate a different prosodic target for carrying out the unit selection (as described in 3). Otherwise, the general form and weights on the objective function were the same for all conditions.

### 6.1. Effect on the Search

To assess the effect of each prosodic-target model on the search, we logged the output of the unit selection when using either model. Altogether, the 40 texts resulted in 12,183 units, excluding silences. Of these, 3,453 are commonly selected by the search when either prosodic target is employed, and the remaining 8,730 units differ in either case. The two prosodic models therefore have a substantial effect on the symbolic output of the search through the target cost component of the objective function, differing 71.66% of the time in their selection. What this amounts to perceptually is something we will investigate through a listening test. An objective metric about the outcome of the search with perhaps more qualitative implications is the splice rate (the percentage of units that are spliced together from non-contiguous parts of the database to generate the output), since outputs with low splice rate contain longer spans selected wholesale from the inventory and tend to better preserve naturalness. However, in this regard, the two systems differ less, with 43.33% and 46.68% splice rates for the BiRNN and Random Forest models respectively (an absolute 3.35% reduction in splice count in favor of BiRNN). Note that these results are independent of the bypass condition used; the bypass setting is only relevant at the post-search waveform-generation stage, which we investigate next.

### 6.2. Perceptual Evaluation

A listening test was designed to perceptually evaluate the effect of the different prosodic models across two separate conditions for the variants of bypass algorithm discussed. Within each condition, we designed a "playlist" of 40 audio samples to present to human listeners for evaluation. We used a round-robin scheme to assign text-and-system combinations within each condition across two mutually exclusive playlists such that: (a) a playlist would contain an equal number of samples from each system (20 generated using a BiRNN prosodic model, and 20 using a Random Forest), and (b) no text would be used more than once within a playlist to avoid habituation effects. After satisfying these constraints, the order of the samples was randomized within each playlist. Subjects used an interface that allowed them to listen to one sample at a time (as many times as they wished) and rated the overall quality on a 5-point scale, defined as: 1=Poor, 2=Bad, 3=Good, 4=Very Good, 5=Excellent. No response could be registered before the audio finished playing at least once.

We recruited 19 subjects, most of whom were native (and all highly proficient) speakers of English, and assigned 10 to the no-bypass condition and 9 to the bypass condition, in order to evaluate the 40 sentences. This resulted in 400 and 360 overall evaluations for each of the no-bypass and bypass conditions, respectively. Table 1 shows the distribution of the scores for each of the two systems evaluated across the two conditions, as well as the mean opinion score (MOS). As we see, the BiRNN models show a lead over the baseline in terms of the perceived overall quality of the samples. The gap is most notable under the No-Bypass condition ($\Delta = 0.29$) and more modest when the Bypass Algorithm is employed ($\Delta = 0.17$). It is interesting to note that the overall score of the BiRNNs is fairly stable across the two conditions (3.461 vs 3.470), whereas the baseline predictions benefit from the use of the Bypass Algorithm (3.180 vs 3.294). The BiRNNs MOS are particularly encouraging because while the Bypass Algorithm can generally provide an additional degree of naturalness by preserving natural prosody over spans of speech, it is also, by its vary nature, constrained to reproduce only (fragments of) contours existing in the database. This points to a second potential weakness: a locally meaningful contour may not be necessarily consistent with the prosody of the full utterance (or beyond). To make TTS more expressive, we are interested in having good dynamic prosodic predictions which reflect the short- and long-term structure of the input (the use of BiRNNs is one step in this direction), and we want to be able to realize such contours even when they cannot be reproduced from existing units in the database (which may necessitate relaxing the use of the Bypass Algorithm).

Table 1: *Results of listening test, showing the distribution of scores on a 5-point scale for each of the models and conditions, total number of responses (N) per system, and mean opinion score (MOS)*

| | | | Scores | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Model | N | 1 | 2 | 3 | 4 | 5 | MOS |
| No Bypass | RF | 200 | 6 | 36 | 89 | 54 | 15 | 3.180 |
| | BiRNN | 200 | 1 | 27 | 74 | 73 | 25 | **3.470** |
| Bypass | RF | 180 | 2 | 43 | 58 | 54 | 23 | 3.294 |
| | BiRNN | 180 | 2 | 30 | 61 | 57 | 30 | **3.461** |

## 7. Conclusions

In this work we have looked at the use of Bidirectional Recurrent Neural Networks as a prosody-prediction model within a unit-selection TTS system. Although these models have been previously used to generate prosodic targets within a parametric synthesis framework, their use has not received the same attention within unit-selection architectures. We have investigated the effect that this model has on the outcome of the unit selection, observing that it led to sequences that differed by more than 70% of the units selected by a baseline system, and with a small splice-rate reduction of 3%. These differences can also be accounted for perceptually, as demonstrated by a listening test that showed a 0.29 absolute MOS score lead over the baseline when the prosodic targets are realized in the output synthesis, and a smaller 0.17 increase when the natural prosody is favored.

## 8. Acknowledgements

The authors would like to thank Larry Sansone for his help running the perceptual listening tests.

## 9. References

[1] R. Fernandez, R. Rendel, B. Ramabhadran, and R. Hoory, "Prosody contour prediction with Long Short-Term Memory, Bi-directional, Deep Recurrent Neural Networks," in *Interspeech*, Singapore, 2014, pp. 2268–2272.

[2] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, "TTS synthesis with Bidirectional LSTM based Recurrent Neural Networks," in *Interspeech*, Singapore, 2014, pp. 1964–1968.

[3] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," in *ICASSP*, 2015 (To Appear).

[4] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using Deep Neural Networks," in *ICASSP*, 2013, pp. 7962–7966.

[5] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *ICASSP*, 2014, pp. 3872–3876.

[6] J. Pitrelli, R. Bakis, E. Eide, R. Fernandez, W. Hamza, and M. Picheny, "The IBM expressive Text-to-Speech synthesis system for American English," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 4, pp. 1099–1108, 2006.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[8] F. A. Gers, J. Schmidhuber, and F. Cummings, "Learning to forget: Continual prediction with LSTM," *Neural Computaiton*, vol. 12, no. 10, pp. 2451–2471, 2000.

[9] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM Recurrent Networks," *J. of Machine Learning Research*, vol. 3, pp. 115–143, 2002.

[10] A. Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, 2012.

[11] A. Graves, M. Abdel-rahman, and G. Hinton, "Speech recognition with Deep Recurrent Neural Networks," in *ICASSP*, 2013, pp. 6885–6889.

[12] 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 expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Jun. 2010.

[13] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[14] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.