HIGH-DIMENSIONAL SEQUENCE TRANSDUCTION

Nicolas Boulanger-Lewandowski *Yoshua Bengio* *Pascal Vincent*

Dept. IRO, Université de Montréal
Montréal (QC), H3C 3J7, Canada

ABSTRACT

We investigate the problem of transforming an input sequence into a high-dimensional output sequence in order to transcribe polyphonic audio music into symbolic notation. We introduce a probabilistic model based on a recurrent neural network that is able to learn realistic output distributions given the input and we devise an efficient algorithm to search for the global mode of that distribution. The resulting method produces musically plausible transcriptions even under high levels of noise and drastically outperforms previous state-of-the-art approaches on five datasets of synthesized sounds and real recordings, approximately *halving the test error rate*.

Index Terms— Sequence transduction, restricted Boltzmann machine, recurrent neural network, polyphonic transcription

1. INTRODUCTION

Machine learning tasks can often be formulated as the transformation, or *transduction*, of an input sequence into an output sequence: speech recognition, machine translation, chord recognition or automatic music transcription, for example. Recurrent neural networks (RNN) [1] offer an interesting route for sequence transduction [2] because of their ability to represent arbitrary output distributions involving complex temporal dependencies at different time scales.

When the output predictions are high-dimensional vectors, such as tuples of notes in musical scores, it becomes very expensive to enumerate all possible configurations at each time step. One possible approach is to capture high-order interactions between output variables using restricted Boltzmann machines (RBM) [3] or a tractable variant called NADE [4], a weight-sharing form of the architecture introduced in [5]. In a recently developed probabilistic model called the RNN-RBM, a series of distribution estimators (one at each time step) are conditioned on the deterministic output of an RNN [6, 7]. In this work, we introduce an input/output extension of the RNN-RBM that can learn to map input sequences to output sequences, whereas the original RNN-RBM only learns the output sequence distribution. In contrast to the approach of [2] designed for discrete output symbols, or one-hot vectors, our high-dimensional paradigm requires a more elaborate inference procedure. Other differences include our use of second-order Hessian-free (HF) [8] optimization[1] but not of LSTM cells [9] and, for simplicity and performance reasons, our use of a single recurrent network to perform both transcription and temporal smoothing. We also do not need special "null" symbols since the sequences are already aligned in our main task of interest: polyphonic music transcription.

The authors would like to thank NSERC, CIFAR and the Canada Research Chairs for funding, and Compute Canada/Calcul Québec for computing resources.

[1]Our code is available online at `http://www-etud.iro.umontreal.ca/~boulanni/icassp2013`.

The objective of polyphonic transcription is to obtain the underlying notes of a polyphonic audio signal as a symbolic *piano-roll*, i.e. as a binary matrix specifying precisely which notes occur at each time step. We will show that our transduction algorithm produces more musically plausible transcriptions in both noisy and normal conditions and achieve superior overall accuracy [10] compared to existing methods. Our approach is also an improvement over the hybrid method in [6] that combines symbolic and acoustic models by a product of experts and a greedy chronological search, and [11] that operates in the time domain under Markovian assumptions. Finally, [12] employs a bidirectional RNN without temporal smoothing and with independent output note probabilities. Other tasks that can be addressed by our transduction framework include automatic accompaniment, melody harmonization and audio music denoising.

2. PROPOSED ARCHITECTURE

2.1. Restricted Boltzmann machines

An RBM is an energy-based model where the joint probability of a given configuration of the visible vector $v \in \{0,1\}^N$ (output) and the hidden vector h is:

$$P(v,h) = \exp(-b_v^{\mathrm{T}} v - b_h^{\mathrm{T}} h - h^{\mathrm{T}} W v)/Z \tag{1}$$

where b_v, b_h and W are the model parameters and Z is the usually intractable partition function. The marginalized probability of v is related to the free-energy $F(v)$ by $P(v) \equiv e^{-F(v)}/Z$:

$$F(v) = -b_v^{\mathrm{T}} v - \sum_i \log(1 + e^{b_h + W v})_i \tag{2}$$

The gradient of the negative log-likelihood of an observed vector v involves two opposing terms, called the positive and negative phase:

$$\frac{\partial(-\log P(v))}{\partial \Theta} = \frac{\partial F(v)}{\partial \Theta} - \frac{\partial(-\log Z)}{\partial \Theta} \tag{3}$$

where $\Theta \equiv \{b_v, b_h, W\}$. The second term can be estimated by a single sample v^* obtained from a Gibbs chain starting at v:

$$\frac{\partial(-\log P(v))}{\partial \Theta} \simeq \frac{\partial F(v)}{\partial \Theta} - \frac{\partial F(v^*)}{\partial \Theta}. \tag{4}$$

resulting in the well-known contrastive divergence algorithm [13].

2.2. NADE

The neural autoregressive distribution estimator (NADE) [4] is a tractable model inspired by the RBM. NADE is similar to a fully visible sigmoid belief network in that the conditional probability distribution of a visible unit v_j is expressed as a nonlinear function of the vector $v_{<j} \equiv \{v_k, \forall k < j\}$:

$$P(v_j = 1 | v_{<j}) = \sigma(W_{:,j}^{\top} h_j + (b_v)_j) \tag{5}$$

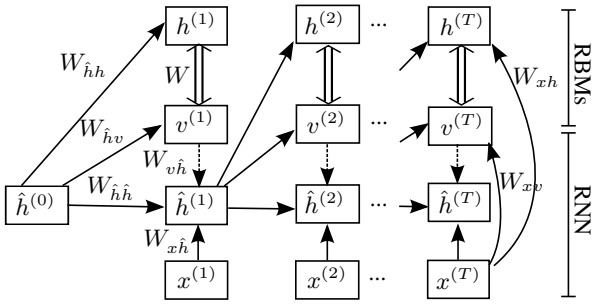

Fig. 1. Graphical structure of the I/O RNN-RBM. Single arrows represent a deterministic function, double arrows represent the hidden-visible connections of an RBM, dotted arrows represent optional connections for temporal smoothing. The $x \to \{v, h\}$ connections have been omitted for clarity at each time step except the last.

$$h_j = \sigma(W_{:,<j} v_{<j} + b_h) \tag{6}$$

where $\sigma(x) \equiv (1 + e^{-x})^{-1}$ is the logistic sigmoid function.

In the following discussion, one can substitute RBMs with NADEs by replacing equation (4) with the exact gradient of the negative log-likelihood cost $C \equiv -\log P(v)$:

$$\frac{\partial C}{\partial (b_v)_j} = P(v_j = 1 | v_{<j}) - v_j \tag{7}$$

$$\frac{\partial C}{\partial b_h} = \sum_{k=1}^{N} \frac{\partial C}{\partial (b_v)_k} W_{:,k} h_k (1 - h_k) \tag{8}$$

$$\frac{\partial C}{\partial W_{:,j}} = \frac{\partial C}{\partial (b_v)_j} h_j + v_j \sum_{k=j+1}^{N} \frac{\partial C}{\partial (b_v)_k} W_{:,k} h_k (1 - h_k) \tag{9}$$

In addition to the possibility of using HF for training, a tractable distribution estimator is necessary to compare the probabilities of different output sequences during inference.

2.3. The input/output RNN-RBM

The I/O RNN-RBM is a sequence of conditional RBMs (one at each time step) whose parameters $b_v^{(t)}, b_h^{(t)}, W^{(t)}$ are time-dependent and depend on the sequence history at time t, denoted $\mathcal{A}^{(t)} \equiv \{x^{(\tau)}, v^{(\tau)} | \tau < t\}$ where $\{x^{(t)}\}, \{v^{(t)}\}$ are respectively the input and output sequences. Its graphical structure is depicted in Figure 1. Note that by ignoring the input x, this model would reduce to the RNN-RBM [6]. The I/O RNN-RBM is formally defined by its joint probability distribution:

$$P(\{v^{(t)}\}) = \prod_{t=1}^{T} P(v^{(t)} | \mathcal{A}^{(t)}) \tag{10}$$

where the right-hand side multiplicand is the marginalized probability of the t^{th} RBM (eq. 2) or NADE (eq. 5).

Following our previous work, we will consider the case where only the biases are variable:

$$b_h^{(t)} = b_h + W_{\hat{h}h} \hat{h}^{(t-1)} + W_{xh} x^{(t)} \tag{11}$$

$$b_v^{(t)} = b_v + W_{\hat{h}v} \hat{h}^{(t-1)} + W_{xv} x^{(t)} \tag{12}$$

where $\hat{h}^{(t)}$ are the hidden units of a single-layer RNN:

$$\hat{h}^{(t)} = \sigma(W_{v\hat{h}} v^{(t)} + W_{\hat{h}\hat{h}} \hat{h}^{(t-1)} + W_{x\hat{h}} x^{(t)} + b_{\hat{h}}) \tag{13}$$

where the indices of weight matrices and bias vectors have obvious meanings. The special case $W_{v\hat{h}} = 0$ gives rise to a transcription network without temporal smoothing. Gradient evaluation is based on the following general scheme:

1. Propagate the current values of the hidden units $\hat{h}^{(t)}$ in the RNN portion of the graph using (13),
2. Calculate the RBM or NADE parameters that depend on $\hat{h}^{(t)}, x^{(t)}$ (eq. 11-12) and obtain the log-likelihood gradient with respect to W, $b_v^{(t)}$ and $b_h^{(t)}$ (eq. 4 or eq. 7-9),
3. Propagate the estimated gradient with respect to $b_v^{(t)}, b_h^{(t)}$ backward through time (BPTT) [1] to obtain the estimated gradient with respect to the RNN parameters.

By setting $W = 0$, the I/O-RNN-RBM reduces to a regular RNN that can be trained with the cross-entropy cost:

$$L(\{v^{(t)}\}) = \frac{1}{T} \sum_{t=1}^{T} \sum_{j=1}^{N} -v_j^{(t)} \log p_j^{(t)} - (1 - v_j^{(t)}) \log(1 - p_j^{(t)}) \tag{14}$$

where $p^{(t)} = \sigma(b_v^{(t)})$ and equations (12) and (13) hold. We will use this model as one of our baselines for comparison.

A potential difficulty with this training scenario stems from the fact that since v is known during training, the model might (understandably) assign more weight to the symbolic information than the acoustic information. This form of *teacher forcing* during training could have dangerous consequences at test time, where the model is autonomous and may not be able to recover from past mistakes. The extent of this condition obviously depends on the ambiguousness of the audio and the intrinsic predictability of the output sequences, and can also be controlled by introducing noise to either $x^{(\tau)}$ or $v^{(\tau)}, \tau < t$, or by adding the regularization terms $\alpha(|W_{xv}|^2 + |W_{xh}|^2) + \beta(|W_{\hat{h}v}|^2 + |W_{\hat{h}h}|^2)$ to the objective function. It is trivial to revise the stochastic gradient descent updates to take those penalties into account.

3. INFERENCE

A distinctive feature of our architecture are the (optional) connections $v \to \hat{h}$ that implicitly tie $v^{(t)}$ to its history $\mathcal{A}^{(t)}$ and encourage coherence between successive output frames, and temporal smoothing in particular. At test time, predicting one time step $v^{(t)}$ requires the knowledge of the previous decisions on $v^{(\tau)}$ (for $\tau < t$) which are yet uncertain (not chosen optimally), and proceeding in a greedy chronological manner does not necessarily yield configurations that maximize the likelihood of the complete sequence[2]. We rather favor a global search approach analogous to the Viterbi algorithm for discrete-state HMMs. Since in the general case the partition function of the t^{th} RBM depends on $\mathcal{A}^{(t)}$, comparing sequence likelihoods becomes intractable, hence our use of the tractable NADE.

Our algorithm is a variant of beam search for high-dimensional sequences, with beam width w and maximal branching factor K (Algorithm 1). Beam search is a breadth-first tree search where only the w most promising paths (or nodes) at depth t are kept for future examination. In our case, a node at depth t corresponds to a subsequence of length t, and all descendants of that node are assumed to share the same sequence history $\mathcal{A}^{(t+1)}$; consequently, only $v^{(t)}$ is allowed to change among siblings. This structure facilitates identifying the most promising paths by their cumulative log-likelihood. For

[2]Note that without temporal smoothing ($W_{v\hat{h}} = 0$), the $v^{(t)}, 1 \leq t \leq T$ would be conditionally independent given x and the prediction could simply be obtained separately at each time step t.

Algorithm 1 HIGH-DIMENSIONAL BEAM SEARCH

Find the most likely sequence $\{v^{(t)}, 1 \leq t \leq T\}$ under a model m with beam width w and branching factor K.

$q \leftarrow$ min-priority queue
q.insert$(0, m)$
for $t = 1 \dots T$ **do**
  $q' \leftarrow$ min-priority queue of capacity w *
  while $l, m \leftarrow q$.pop() **do**
    for l', v' **in** m.find_most_probable(K) **do**
      $m' \leftarrow m$ with $v^{(t)} := v'$
      q'.insert$(l + l', m')$
  $q \leftarrow q'$
return q.pop()

*A *min*-priority queue of fixed capacity w maintains (at most) the w *highest* values at all times.

high-dimensional output however, any non-leaf node has exponentially many children (2^N), which in practice limits the exploration to a fixed number K of siblings. This is necessary because enumerating the configurations at a given time step by decreasing likelihood is intractable (e.g. for RBM or NADE) and we must resort to stochastic search to form a pool of promising children at each node. Stochastic search consists in drawing S samples of $v^{(t)}|A^{(t)}$ and keeping the K unique most probable configurations. This procedure usually converges rapidly with $S \simeq 10K$ samples, especially with strong biases coming from the conditional terms. Note that $w = 1$ or $K = 1$ reduces to a greedy search, and $w = 2^{NT}, K = 2^N$ corresponds to an exhaustive breadth-first search.

When the output units $v_j^{(t)}, 0 \leq j < N$ are conditionally independent given $\mathcal{A}^{(t)}$, such as for a regular RNN (eq. 14), it is possible to enumerate configurations by decreasing likelihood using a dynamic programming approach (Algorithm 2). This very efficient algorithm in $O(K \log K + N \log N)$ is based on linearly growing priority queues, where K need not be specified in advance. Since inference is usually the bottleneck of the computation, this optimization makes it possible to use much higher beam widths w with unbounded branching factors for RNNs.

Algorithm 2 INDEPENDENT OUTPUTS INFERENCE

Enumerate the K most probable configurations of N independent Bernoulli random variables with parameters $0 < p_i < 1$.

$v_0 \leftarrow \{i : p_i \geq 1/2\}$
$l_0 \leftarrow \sum_i \log(\max(p_i, 1 - p_i))$
yield l_0, v_0
$L_i \leftarrow |\log \frac{p_i}{1-p_i}|$
sort L, store corresponding permutation R
$q \leftarrow$ min-priority queue
q.insert$(L_0, \{0\})$
while $l, v \leftarrow q$.pop() **do**
  yield $l_0 - l$, $v_0 \triangle R[v]$ *
  $i \leftarrow \max(v)$
  if $i + 1 < N$ **then**
    q.insert$(l + L_{i+1}, v \cup \{i + 1\})$
    q.insert$(l + L_{i+1} - L_i, v \cup \{i + 1\} \setminus \{i\})$

$^\star A \triangle B \equiv (A \cup B) \setminus (A \cap B)$ denotes the symmetric difference of two sets. $R[v]$ indicates the R-permutation of indices in the set v.

A pathological condition that sometimes occurs with beam search over long sequences ($T \gg 200$) is the exponential duplication of highly likely quasi-identical paths differing only at a few time steps, that quickly saturate beam width with essentially useless variations. Several strategies have been tried with moderate success in those cases, such as committing to the most likely path every M time steps (*periodic restarts* [14]), pruning similar paths, or pruning paths with identical τ previous time steps (the *local assumption*), where τ is a maximal time lag that the chosen architecture can reasonably describe (e.g. $\tau \simeq 200$ for RNNs trained with HF). It is also possible to initialize the search with Algorithm 1 then backtrack at each node iteratively, resulting in an anytime algorithm [15].

Dataset	HMM [16]	RNN-RBM [6]	Proposed
Piano-midi.de	59.5%	60.8%	64.1%
Nottingham	71.4%	77.1%	97.4%
MuseData	35.1%	44.7%	66.6%
JSB Chorales	72.0%	80.6%	91.7%

Table 1. Frame-level transcription accuracy obtained on four datasets by the Nam et al. algorithm with HMM temporal smoothing [16], using the RNN-RBM musical language model [6], or the proposed I/O RNN-NADE model.

4. EXPERIMENTS

In the following experiments, the acoustic input $x^{(t)}$ is constituted of powerful DBN-based learned representations [16]. The magnitude spectrogram is first computed by the short-term Fourier transform using a 128 ms sliding Blackman window truncated at 6 kHz, normalized and cube root compressed to reduce the dynamic range. We apply PCA whitening to retain 99% of the training data variance, yielding roughly 30–70% dimensionality reduction. A DBN is then constructed by greedy layer-wise stacking of sparse RBMs trained in an unsupervised way to model the previous hidden layer expectation ($v^{l+1} \equiv \mathbb{E}[h^l|v^l]$) [17]. The whole network is finally finetuned with respect to a supervised criterion (e.g. eq. 14) and the last layer is then used as our input $x^{(t)}$ for the spectrogram frame at time t.

We evaluate our method on five datasets of varying complexity: Piano-midi.de, Nottingham, MuseData and JSB chorales (see [6]) which are rendered from piano and orchestral instrument soundfonts, and Poliner & Ellis [18] that comprises synthesized sounds and real recordings. We use frame-level accuracy [10] for model evaluation. Hyperparameters are selected by a random search [19] on predefined intervals to optimize validation set accuracy; final performance is reported on the test set.

Table 1 compares the performance of the I/O RNN-RBM to the HMM baseline [16] and the RNN-RBM hybrid approach [6] on four datasets. Contrarily to the product of experts of [6], our model is jointly trained, which eliminates duplicate contributions to the energy function and the related increase in marginals temperature, and provides much better performance on all datasets, approximately halving the error rate in average over these datasets.

We now assess the robustness of our algorithm to different types of noise: white noise, pink noise, masking noise and spectral distortion. In masking noise, parts of the signal of exponentially distributed length ($\mu = 0.4$ s) are randomly destroyed [20]; spectral distortion consists in Gaussian pitch shifts of amplitude σ [21]. The first two types are simplest because a network can recover from them by averaging neighboring spectrogram frames (e.g. Kalman smoothing), whereas the last two time-coherent types require higher-level musical understanding. We compare a bidirectional RNN [12] adapted for frame-level transcription, a regular RNN with $v \rightarrow \hat{h}$ connections ($w = 2000$) and the I/O RNN-NADE ($w = 50, K = 10$). Figure 2 illustrates the importance of temporal smoothing con-

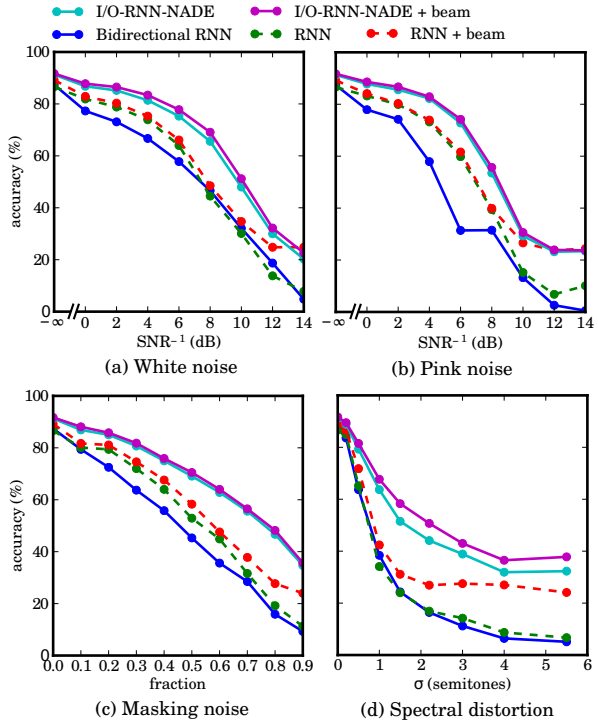

Fig. 2. Robustness to different types of noise of various RNN-based models on the JSB chorales dataset.

SONIC [22]	39.6%
Note events + HMM [23]	46.6%
Linear SVM [18]	67.7%
DBN + SVM [16]	72.5%
BLSTM RNN [12]	75.2%
AdaBoost cascade [24]	75.2%
I/O-RNN-NADE	79.1%

Table 2. Frame-level accuracy of existing transcription methods on the Poliner & Ellis dataset [18].

nections and the additional advantage provided by conditional distribution estimators. Beam search is responsible for a 0.5% to 18% increase in accuracy over a greedy search ($w = 1$).

Figure 3 shows transcribed piano-rolls for various RNNs on an excerpt of Bach's chorale *Es ist genug* with 6 dB pink noise (Fig. 3(a)). We observe that a bidirectional RNN is unable to perform temporal smoothing on its own (Fig. 3(b)), and that even a post-processed version (Fig. 3(c)) can be improved by our global search algorithm (Fig. 3(d)). Our best model offers an even more musically plausible transcription (Fig. 3(e)). Finally, we compare the transcription accuracy of common methods on the Poliner & Ellis [18] dataset in Table 2, that highlights impressive performance.

5. CONCLUSIONS

We presented an input/output model for high-dimensional sequence transduction in the context of polyphonic music transcription. Our model can learn basic musical properties such as temporal continuity, harmony and rhythm, and efficiently search for the most musically plausible transcriptions when the audio signal is partially destroyed, distorted or temporarily inaudible. Conditional distribution estimators are important in this context to accurately describe the

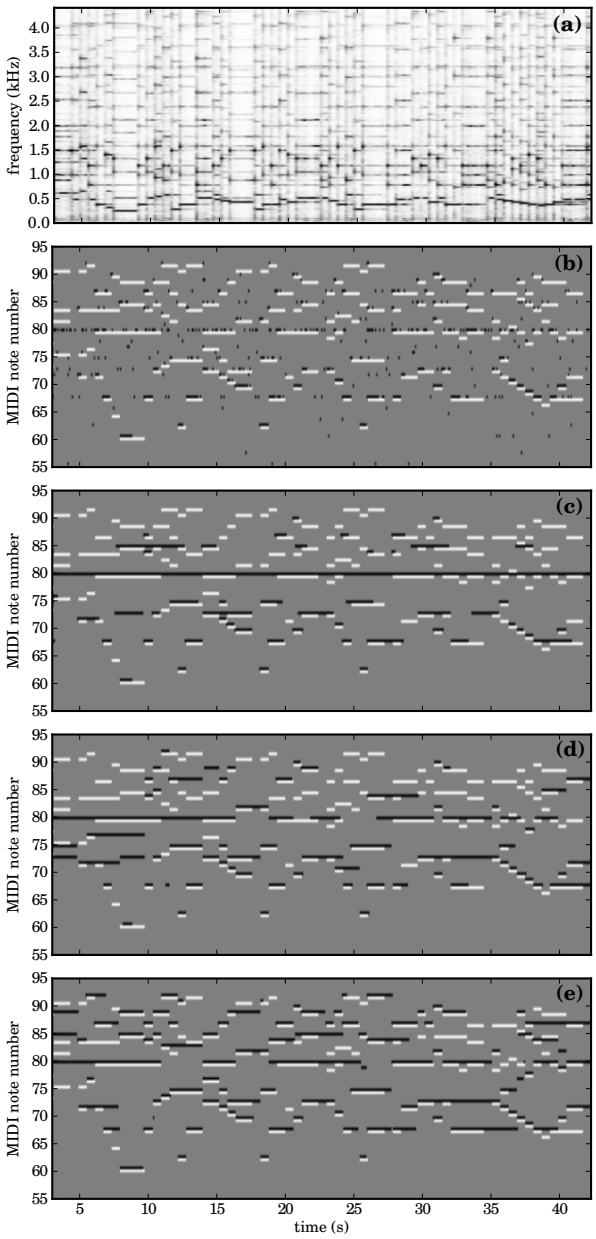

Fig. 3. Demonstration of temporal smoothing on an excerpt of Bach's chorale *Es ist genug* (BWV 60.5) with 6 dB pink noise. Figure shows (a) the raw magnitude spectrogram, and transcriptions by (b) a bidirectional RNN, (c) a bidirectional RNN with HMM post-processing, (d) an RNN with $v \to \hat{h}$ connections ($w = 75$) and (e) I/O-RNN-NADE ($w = 20$, $K = 10$). Predicted piano-rolls (black) are interleaved with the ground-truth (white) for comparison.

density of *multiple* potential paths given the weakly discriminative audio. This ability translates well to the transcription of "clean" signals where instruments may still be buried and notes occluded due to interference, ambient noise or imperfect recording techniques. Our algorithm approximately halves the error rate with respect to competing methods on five polyphonic datasets based on frame-level accuracy. Qualitative testing also suggests that a more musically relevant metric would enhance the advantage of our model, since transcription errors often constitute reasonable alternatives.

6. REFERENCES

[1] D.E. Rumelhart, G.E. Hinton, and R.J. Williams, "Learning internal representations by error propagation," in *Parallel Dist. Proc.*, pp. 318–362. MIT Press, 1986.

[2] A. Graves, "Sequence transduction with recurrent neural networks," in *ICML 29*, 2012.

[3] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Dist. Proc.*, pp. 194–281. MIT Press, 1986.

[4] H. Larochelle and I. Murray, "The neural autoregressive distribution estimator," *JMLR: W&CP*, vol. 15, pp. 29–37, 2011.

[5] Y. Bengio and S. Bengio, "Modeling high-dimensional discrete data with multi-layer neural networks," in *NIPS 12*, 2000, pp. 400–406.

[6] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in *ICML 29*, 2012.

[7] I. Sutskever, G. Hinton, and G. Taylor, "The recurrent temporal restricted Boltzmann machine," in *NIPS 20*, 2008, pp. 1601–1608.

[8] J. Martens and I. Sutskever, "Learning recurrent neural networks with Hessian-free optimization," in *ICML 28*, 2011.

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

[10] M. Bay, A.F. Ehmann, and J.S. Downie, "Evaluation of multiple-F0 estimation and tracking systems," in *ISMIR*, 2009.

[11] A.T. Cemgil, H.J. Kappen, and D. Barber, "A generative model for music transcription," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 2, pp. 679–694, 2006.

[12] S. Böck and M. Schedl, "Polyphonic piano note transcription with recurrent neural networks," in *ICASSP*, 2012, pp. 121–124.

[13] G.E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[14] S. Richter, J.T. Thayer, and W. Ruml, "The joy of forgetting: Faster anytime search via restarting," in *ICAPS*, 2010, pp. 137–144.

[15] R. Zhou and E.A. Hansen, "Beam-stack search: Integrating backtracking with beam search," in *ICAPS*, 2005, pp. 90–98.

[16] J. Nam, J. Ngiam, H. Lee, and M. Slaney, "A classification-based polyphonic piano transcription approach using learned feature representations," in *ISMIR*, 2011.

[17] Y. Bengio, "Learning deep architectures for AI," *Foundations and Trends in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[18] G.E. Poliner and D.P.W. Ellis, "A discriminative model for polyphonic piano transcription," *JASP*, vol. 2007, no. 1, pp. 154–164, 2007.

[19] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, 2012.

[20] P. Vincent, H. Larochelle, Y. Bengio, and P.A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML 25*, 2008, pp. 1096–1103.

[21] K.J. Palomäki, G.J. Brown, and J.P. Barker, "Techniques for handling convolutional distortion with 'missing data' automatic speech recognition," *Speech Communication*, vol. 43, no. 1, pp. 123–142, 2004.

[22] M. Marolt, "A connectionist approach to automatic transcription of polyphonic piano music," *IEEE Transactions on Multimedia*, vol. 6, no. 3, pp. 439–449, 2004.

[23] M.P. Ryynänen and A. Klapuri, "Polyphonic music transcription using note event modeling," in *ASPAA*, 2005, pp. 319–322.

[24] C.G. Boogaart and R. Lienhart, "Note onset detection for the transcription of polyphonic piano music," in *ICME*, 2009, pp. 446–449.