The LIMSI Handwriting Recognition System for the HTRtS 2014 Contest

Théodore Bluche*‡, Hermann Ney *† and Christopher Kermorvant‡§
*LIMSI CNRS, Spoken Language Processing Group, Orsay, France
†RWTH Aachen University, Human Language Technology and Pattern Recognition, Aachen, Germany
‡A2iA SA, Paris, France
§Teklia SAS, Paris, France

***Abstract*—In this paper we present the handwriting recognition systems submitted by the LIMSI to the HTRtS 2014 contest. The systems for both the restricted and unrestricted tracks consisted of combination of several optical models. We extracted handcrafted features as well as pixels values with a sliding window. We trained Deep Neural Networks (DNNs) and Bidirectional Long Short-Term Memory Recurrent Neural Networks (BLSTM-RNNs), which where plugged as the optical model in Hidden Markov Models (HMMs). We propose a novel method to build language models that can cope with hyphenation in the text. The combination was performed from lattices generated from the different systems. We were the only team participating in both tracks and ranked second in each. The final Word Error Rates were 15.0% and 11.0% for the restricted (*resp.* unrestricted) track. We studied the impact of adding data for optical and language modeling. After the evaluation, we also used the same corpus for the language model as the winning team and obtained comparable results.**

I. Introduction

The HTRtS contest was held in the scope of the International Conference on Frontiers in Handwriting Recognition (ICFHR 2014) [1]. The documents, prepared for the tranScriptorium project by the University College, London, consist of a subset of the Bentham collection. They are handwritten manuscripts written by the British philosopher Jeremy Betham and his staff in the 18th and 19th centuries.

For this contest, we built several systems, using the provided data, as well as additional data. We trained hybrid Neural Networks (NN) / HMM systems, with different neural networks, namely deep NNs and recurrent NNs, working on different types of features: handcrafted and pixel values. For improved performance, we also combined the different systems.

We propose a very simple method to cope with the numerous hyphenations in this database, which cause an increase of out-of-vocabulary word rate. We were the only participant to take part in both evaluation tracks. In the *restricted* track, only the provided data are allowed to train the systems. In the *unrestricted* track, participants can use additional data. We studied the effect of adding more data for the optical and language models. Unsurprisingly, both brought significant improvements, although those obtained with a better language model are easier to get. The final Word Error Rates (WERs) were 15.0% and 11.0% for the restricted (*resp.* unrestricted) track, which ranked us second in both.

This paper gives the details of our systems, and is divided as follows. In Section II, we introduce the data we used to train the systems. In Section III, we present the different components of the systems. In Section IV, we analyze the results obtained. Finally, we describe additional experiments carried out after the evaluation in Section V, and conclude in Section VI.

II. Data

The provided training data are images from Jeremy Bentham personal notes, written by himself and his staff in English, around the 18th and 19th centuries. It consists of 350 pages (9,198 lines). The validation set comprises 50 images, or 1,415 lines. The test set is made of 33 pages, or 860 lines. We used the training set annotations to build the language models, which consists of more than 76k running words (12k unique words), built from 92 different characters.

For the unrestricted track, we were allowed to use additional data to train the models. We gathered images from five different databases, corresponding to English and/or ancient texts. Some of them were not annotated at the line level, so we used an automatic method to create the training data.

The IAM database [2] is a well-known database for handwriting recognition. The training set consists of 747 pages of clean handwritten passages of the English LOB corpus, copied by different writers. The positions and transcripts of text lines are provided with the database.

The Georges Washington database [3] is extracted from Washington Papers, dating from the 18th century. It is written in English by two writers, and the available version provided by the University of Bern consists of preprocessed text lines along with their transcription.

The NUMEN database, provided by Numen Digital, comprises 13,649 historical documents in old creole French containing land surveying reports, with annotations of line positions and transcriptions. We only used a subset of 11,710 lines for training.

The IBM UB 1 database [4] was collected at the University of Buffalo. It contains online and offline data, totalling around 6,000 pages of cursive handwritten texts produced by more than 40 writers. The line positions are unknown, and the transcript is provided only for whole pages or couples of pages.

The Abraham Lincoln database [5] is not an official database. It was retrieved from the Library of Congress, and is

made of Lincoln's correspondence manuscript documents from the 19th century. The documents were produced by different writers. The line positions are unknown and the transcript is available only for whole documents of several pages.

We extracted subsets of text lines with mapped transcript from the last two databases using the method presented in [6]. We modified the algorithm parameters to retrieve only those lines for which we can be confident that the extracted position and mapped transcript are correct.

A summary of the data used for training the optical models is presented on Table I. Later, when we say that we use only a subset of some database, it corresponds to approximately 10% of randomly selected lines.

TABLE I: Data used for optical model training.

Name	Number of text lines
Restricted	
Bentham	9,198
Unrestricted	
IAM	6,482
NUMEN	11,710
G. Washington (GW)	642
IBM UB 1	825
A. Lincoln (AL)	3,960

For the unrestricted track, we added the Open American National Corpus [7] to the estimation of the language models. We used only the part extracted from written documents, which amounts to more than 11M words.

III. Description of the System Components

A. Preprocessing

The systems are trained with text lines cropped from the whole 300 DPI document images. They are first converted to grey-level. Then, the lines are deslanted using the approach described in [8]. The contrast is enhanced by mapping the 5% darkest pixels to black and 70% lightest ones to white with a linear interpolation in between. Twenty white pixels are added on left and right to account for empty context. All line images were normalized in height to 72px.

B. Feature Extraction

We used two kinds of features: handcrafted features, and raw pixel intensities. A sliding window is scanned through the line image to extract features. The handcrafted features [9] are geometrical and statistical features (pixel densities, foreground/background transitions, position of the center of gravity, counts of pixels configurations, plus their derivatives), extracted with a sliding window of width 3px with a shift of 3px, resulting in 56-dimensional feature vectors. The pixel intensities are extracted with a sliding window of width 57px and 3px shift, rescaled to 25x32px, producing 800-dimensional feature vectors.

C. Character/Lexical Modeling

The different optical models described below model all 92 unicode characters present in the training data provided for the competition, plus the whitespace. In the case of RNNs, a special symbol is also modeled to represent anything that is not a character prediction. This *blank* symbol is taken into account in the lexicon. It is optional between any two different characters, and mandatory between two characters which are the same (*e.g.* between the *t*'s of *little*).

D. Gaussian Mixture Model - Hidden Markov Models (GMM-HMMs)

We first trained a GMM-HMM system for bootstrapping the training of Neural Networks. The character HMMs were 6-state left-to-right models: each state has a transition to itself and to the next state. Furthermore, two two-state whitespace HMMs were also built to model the whitespaces at the beginning and at the end of words. The observations for these HMMs are the handcrafted features. The GMM-HMM is trained following the standard Expectation-Maximization (EM) procedure with a Maximum Likelihood criterion with the Kaldi toolkit [10]. At each iteration, the total number of Gaussians is increased, and we stop the training when the WER on the validation data stops decreasing. The final model has 67,037 diagonal-covariance Gaussians in 556 densities (one density per HMM state). The results on the validation set are 27.9% WER and 14.5% Character Error Rate (CER), using the restricted language model described in Section III-G.

E. Deep Neural Networks (DNNs)

The forced alignments computed with the GMM-HMM are used to create a labeled training set for DNNs. We trained DNNs with either handcrafted or pixel features. The networks are pre-trained with the unsupervised layerwise training method described in [11], and fine-tuned with cross-entropy training and stochastic gradient descent.

The inputs are normalized to have zero mean and unit variance on the training data. Each hidden layer has 1,024 units and a sigmoid activation. The output layer has one node for each HMM state, *i.e.* 556. The learning rate is set to 0.008. At each epoch, we compute the frame accuracy of the network over the validation set, and start halving the learning rate for each following epoch as soon as the improvement is below 0.01%. When the accuracy on the validation set stops increasing, the training stops.

We trained networks with different depths, and in the case of handcrafted features, different contexts (number of consecutive frames concatenated to build the network input). The WERs on the validation set (without *word insertion penalty*) are summarized on Table II.

TABLE II: Word Error Rates of DNNs trained with cross-entropy, with different number of hidden layers and different inputs. on the validation set. The best systems are indicated in bold face.

		Number of hidden layers						
Features	**Context**	1	2	3	4	5	6	7
Hand-crafted	$\pm$ 1	27.2	26.4	25.9	26.3	25.5	25.7	25.5
	$\pm$ 3	26.2	25.9	26.3	26.2	26.0	26.2	25.7
	$\pm$ 5	27.7	26.3	25.8	26.0	25.7	25.7	25.6
	$\pm$ 7	27.7	27.2	26.0	26.2	25.7	26.1	25.8
	$\pm$ 9	26.5	25.4	25.1	**24.4**	24.5	24.7	24.6
Pixels	-	33.2	25.0	24.4	23.5	23.8	**22.8**	22.9

We further trained the best networks, in bold face in Table II, with a sequence-discriminative training criterion (state-level Minimum Bayes Risk – sMBR [12], [13]), after realignment of the training set using the cross-entropy-trained networks. We record relative WER improvements up to 11.9% (Table III).

TABLE III: Improvement brought by sMBR sequence training on the validation set

Features	WER	CER
Handcrafted	21.0%	8.9%
+ sMBR training	19.4% (-7.6%)	7.9% (-11.2%)
Pixels	22.6%	10.7%
+ sMBR training	19.9% (-11.9%)	8.2% (-23.4%)

We did not train or fine-tune these network using other data than the one in the official training set. In the following we will refer to the two selected networks as:

- **DNN features** : sMBR-trained DNN using ± 9 input frames of handcrafted features, with 4 hidden layers of 1,024 nodes.
- **DNN pixels** : sMBR-trained DNN using pixel features, with 6 hidden layers of 1,024 nodes.

F. Recurrent Neural Networks (RNNs)

For both types of inputs – handcrafted features and pixels, we trained BLSTM-RNNs. The architectures alternate LSTM layers, with recurrent connections in both directions, and feed-forward layers. Hence, an architecture of 3×100 in Table IV has three layers. The first one has two (parallel) LSTM layers (forward and backward) with 100 units, followed by a feed-forward layer. The third hidden layer, made of forward and backward LSTM layers, is connected to the output layer of the network.

The inputs are the sequences of feature vectors, and there is one output for each character, plus one for a special non-character symbol, used to cope with the different sizes of the input and the output (94 outputs). We trained the networks with the Connectionist Temporal Classification (CTC) objective [14], using stochastic gradient descent and a learning rate of 0.001, to minimize the sequence Negative Log-Likelihood (NLL). After each epoch the NLL is computed for the validation data. The training stops if it did not decrease for 20 epochs, and the best model is kept. To improve the performance of the RNNs, we also applied the dropout technique [15], adapted to RNNs in [16].

The results are presented on Table IV. As previously, the systems we kept are indicated in bold face. In the following, we refer to these systems as:

- **RNN features** : CTC-trained BLSTM-RNN using dropout, handcrafted features, with 7 hidden layers of 200 nodes.
- **RNN pixels** : CTC-trained BLSTM-RNN using dropout, pixel features, with 7 hidden layers of 200 nodes.

We also trained BLSTM-RNNs with additional data for the unrestricted track. Due to time constraints we only trained such systems with handcrafted features, and with the same architecture (7x200 hiddens) and training procedure (CTC, dropout, early stopping, learning rate) as the *RNN features* system.

TABLE IV: RNNs on handcrafted and pixel features (results on the validation set, R-CER is the CER of the RNN alone, without LM).

	Handcrafted Features			Pixels		
	R-CER	WER	CER	R-CER	WER	CER
1x100	17.3	20.6	8.8	38.9	33.8	19.6
3x100	12.8	18.5	7.5	17.7	22.6	10.2
5x100	12.0	19.0	7.6	14.0	20.8	8.7
5x200	11.8	19.9	7.7	14.0	21.4	8.9
7x100	11.1	18.5	7.5	12.2	21.4	8.8
7x200	11.0	18.0	7.0	11.8	20.6	8.4
7x200 + dropout	8.9	**17.2**	6.7	9.2	**18.7**	7.3
9x100	10.6	18.3	7.3	12.1	21.6	8.9

We built three such systems, using different – but not disjoint – subsets of the available images. The training data and results of these systems with the restricted-track language model are presented on Table V.

TABLE V: BLSTM-RNN unrestricted results (RNN-CER is the Character Error Rate with RNN alone, while the WER is after adding the lexicon and LM ; GW and AL stand for G. Washington and A. Lincoln. *"s-"* indicates that only a subset was used).

Name	Training data	RNN-CER	WER
uRNN1	Bentham, GW, sIAM, sNumen, sAL	7.5	16.5
uRNN2	Bentham, GW, sIAM, sNumen, sIBM, sAL	6.6	15.8
uRNN3	Bentham, GW, IAM, Numen, IBM, AL	6.6	15.8

G. Language Modeling (LM)

1) Data preparation: For the restricted track, we first extracted complete paragraphs of text from the line annotations. It makes more sense to build a language model from sentences or whole paragraphs rather than lines or text, because the line boundaries do not necessarily correspond to boundaries in the language. The hyphenation does not make sense at paragraph level, so we reconstructed whole words from hyphenated ones. Next, we normalized the obtained corpora. We split the punctuation from words (this is consistent with the format of the official reference). To limit the size of the vocabulary, we also isolated currency symbols, and split sequences of digits and capital letters.

2) Dealing with hyphenation: To deal with hyphenation, we first generated unigram counts, and recorded all words with more than ten occurrences. We used Pyphen [17] to generate all possible hyphenations of these words. Pyphen produces a list of pairs *(beginning, end)*, so we added the three possible hyphenation symbols at the end (*resp.* beginning) of words beginnings (*resp.* endings), and included them in the final LMs as unigrams with count 1.

3) LM estimation: Finally, we estimated the LMs with the generated counts, and applied Witten-Bell [18] smoothing. For the restricted track, we generated 4grams. For the unrestricted track, we added the Open American National Corpus

(OANC) [7] to the training data, with the same tokenization, and generated bigrams for decoding and trigrams for rescoring. The only difference in the process is that we removed all words with single counts, and then added hyphenations only for words with count higher than 100. We used the SRILM toolkit [19] to estimate the language models.

In the following, we will refer to them as **restricted LM**, and **unrestricted LM**. When using the latter, we decoded with the bigram and rescore the obtained lattices with trigram. Table VI summarizes the vocabulary size, and Out-of-Vocabulary words rate (OOV) and perplexity (PPL) computed on the tokenized validation data. Despite the augmented perplexity of the hyphenated language models, the drop of OOV rate produces a reduction of the transcription error rate.

TABLE VI: LM properties on the validation set. Lines marked with * are the LMs kept for the final systems.

ngram	Training data	$\|V\|$	OOV	PPL	WER
Restricted					
4gram	Bentham	7,318	7.08	98	18.4
4gram-hyph *	Bentham	32,692	5.55	101	17.2
Unrestricted					
2gram	Bentham + OANC	80,744	3.84	318	-
2gram-hyph *	Bentham + OANC	107,780	2.48	345	-
3gram	Bentham + OANC	80,744	3.84	224	-
3gram-hyph *	Bentham + OANC	107,780	2.48	245	-

H. Decoding and Combination

The decoding is performed at line level with the Kaldi toolkit [10]. It is based on a beam search in a weighted Finite-State Transducer comprising the HMM, lexicon, and LM, as described in [20]. It is a hybrid setup [21], where the optical score is given by the NN posteriors $p(s|x)$, divided by the scaled state priors $p(s)^{\alpha}$. For DNNs, the HMMs are the same as for the GMM-HMM system, while the RNNs predicts characters and blanks directly, so we built single-state HMMs to fit the hybrid scheme. Lattices are generated [22] with a fixed beam and optical scale, and rescored with a word insertion penalty, and for the unrestricted track, a higher-order LM. The different systems use different optical modeling and input features, and make different mistakes. We tried two system combination methods to improve the results: the ROVER combination described in [23] and the lattice combination method described in [24]. We combined four systems (DNN/RNN on features/pixels) for the restricted track, and report the results on Table VII, showing the superior performance of the lattice-based combination.

TABLE VII: Comparison of combination techniques for the four restricted track systems.

Method	WER	CER
ROVER combination	16.0	6.6
Lattice combination	15.4	5.9

IV. Result analysis

In this section, we summarize the results presented previously, to show the effect of different aspects of the whole systems. The importance of dropout for RNNs, and sequence training for DNNs have already been shown in Sections III-E and III-F.

In Table VIII we report again the performance of restricted systems. We see that NN optical models bring a huge improvement over the standard ML-trained GMMs. The handcrafted features seem better than raw pixel values, although the difference is small for DNNs. The significant improvement yielded by the combination shows that the systems are complementary.

TABLE VIII: Summary of results of restricted systems on the validation set.

System	WER	CER
GMM-HMM	27.9	14.5
RNN features	17.2	6.7
RNN pixels	18.7	7.3
DNN features	19.4	7.9
DNN pixels	19.9	8.2
Lattice combination of last four	15.4	5.9

Adding more data for the language model estimation seems critical to solve this task, as shown in Table IX (*"Restricted"* versus *"Unrestricted"* columns). We observe relative WER (*resp.* CER) improvements between 12 and 19% (*resp* 8 and 17%).

TABLE IX: Improvements brought by adding more LM data (WER on the validation set). The *unrestricted 2* LM is described in Section V.

	Restricted	Unrestricted	Unrestricted 2
RNN features	17.2	14.9	13.1
RNN pixels	18.7	16.3	14.4
DNN features	19.4	16.7	14.0
DNN pixels	19.9	17.5	14.6
Lattice combination	15.4	12.5	10.7
uRNN1	16.5	13.4	11.9
uRNN2	15.8	13.1	11.3
uRNN3	15.8	13.1	11.3
Lattice combination	14.6	11.8	9.7

Table IX also presents the effect of increasing the amount of training data for optical models. *uRNN* systems should be compared to the *RNN features* system, which have the same inputs and architecture, but only less training data. With both language models, we observe significant WER and CER improvements, and the combination results show that these systems make different errors too.

V. Competition Results and Complementary Experiments

The competition results are presented on Table X. In the official publication [1], only the best results among submitted systems are reported. They correspond to the lattice combinations in each track. Only one other team participated to each track, each using LSTM networks. We ran several complementary experiments. We studied the effect of the corpora used to train the language models. We complete the previous results (Section IV, Table IX) with a new language model. We built it with the same technique as the one described in Section III-G, but using the same corpus as the A2iA system. It is composed of additional Betham texts retrieved from the web [25]. The new language model is referred to as **unrestricted LM 2**. It has a perplexity of 215 and an OOV

TABLE X: Results on the evaluation set. The winner of the official evaluation is given at the bottom of each table.

(a) Restricted track

Model	WER
DNN pixels	20.0
DNN features	19.0
RNN pixels	19.0
RNN features	17.1
Combination	15.0
CITlab	**14.6**

(b) Unrestricted track

Model	WER
RNN features	14.7
uRNN1	12.9
uRNN2	12.7
uRNN3	12.4
Combination	11.1
A2iA	**8.6**

(c) Complementary results

Model	WER
DNN pixels	14.3
DNN features	13.1
RNN pixels	13.7
RNN features	12.4
Combination	10.2
uRNN1	11.1
uRNN2	10.5
uRNN3	10.4
Combination	8.9
A2iA	**8.6**

rate of 1.47% on the validation set. As we can see on Table IX, this new LM seems significantly better. On the evaluation data, our combined system is almost as good as A2iA's, even though they used an open-vocabulary approach, which enable their system to recognize any word (Table Xc).

VI. Conclusion

In this paper, we described the recognition systems submitted to the HTRtS contest in ICFHR 2014, and the key aspects to achieve the final results. The most important one was to use discriminative optical models, which largely outperform GMMs. We trained DNNs and RNNs, and showed that dropout and sequence discriminative training yielded interesting improvements. Concerning language models, hyphenation should be taken care of for this database, since it occurs a lot. Including hyphenation of frequent words to the dictionary increased a lot the vocabulary size, but decreased the OOV rate and the WER. For this task, adding data for language modeling is also highly beneficial, and easier than adding data for optical model training, although this also brought significant improvements. Finally, different recognition systems have been trained. They make different mistakes, and their combination yielded another performance increase. The complementary experiments showed the importance to have a language model trained on relevant data (here, texts by Bentham). With the new LM, our system yielded comparable results to those of the winning team.

Acknowledgment

This work was partially funded by the French National Research Agency (ANR) through the ORIFLAMMS project (ANR-12-CORP-0010).

References

[1] J. A. Sánchez, V. Romero, A. Toselli, and E. Vidal, "ICFHR 2014 HTRtS: Handwritten Text Recognition on tranScriptorium Datasets," in *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2014.

[2] U.-V. Marti and H. Bunke, "The IAM-database: an English sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, Nov. 2002.

[3] A. Fischer, A. Keller, V. Frinken, and H. Bunke, "Lexicon-free handwritten word spotting using character hmms," *Pattern Recognition Letters*, vol. 33, no. 7, pp. 934–942, 2012.

[4] V. Govindaraju, "Ibm-ub data set." [Online]. Available: http://www.cubs.buffalo.edu/hwdata

[5] "Abraham lincoln correspondence." [Online]. Available: http://memory.loc.gov/mss/mal/

[6] T. Bluche, B. Moysset, and C. Kermorvant, "Automatic line segmentation and ground-truth alignment of handwritten documents," in *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2014.

[7] N. Ide and K. Suderman, "The open american national corpus (oanc)," 2007. [Online]. Available: http://www.americannationalcorpus.org/OANC/index.html

[8] R. Buse, Z. Q. Liu, and T. Caelli, "A structural and relational approach to handwritten word recognition." *IEEE Transactions on Systems, Man and Cybernetics*, vol. 27, no. 5, pp. 847–61, Jan. 1997. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/18263093

[9] A.-L. Bianne, F. Menasri, R. Al-Hajj, C. Mokbel, C. Kermorvant, and L. Likforman-Sulem, "Dynamic and Contextual Information in HMM modeling for Handwriting Recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 2066 – 2080, 2011.

[10] D. Povey, A. Ghoshal, N. Goel, M. Hannemann, Y. Qian, P. Schwarz, J. Silovsk, and P. Motl, "The Kaldi Speech Recognition Toolkit," in *Workshop on Automatic Speech Recognition and Understandin*, 2011.

[11] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[12] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *IEEE International Conference on Acoustics, Speech and Signal Processing, 2009. ICASSP 2009.* IEEE, 2009, pp. 3761–3764.

[13] K. Veselý, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Interspeech*, no. 1, 2013, pp. 3–7.

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *International Conference on Machine learning*, 2006, pp. 369–376.

[15] G. Hinton and N. Srivastava, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv: ...*, pp. 1–18, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[16] V. Pham, T. Bluche, C. Kermorvant, and J. Louradour, "Dropout improves recurrent neural networks for handwriting recognition," pp. 285–290, 2014.

[17] W. Berendsen, "Pyphen." [Online]. Available: http://pyphen.org/

[18] I. H. Witten and T. Bell, "The zero-frequency problem: Estimating the probabilities of novel events in adaptive text compression," *Information Theory, IEEE Transactions on*, vol. 37, no. 4, pp. 1085–1094, 1991.

[19] A. Stolcke, "SRILM – An Extensible Language Modeling Toolkit," in *International Conference on Spoken Language Processing*, 2002.

[20] M. Mohri, F. Pereira, and M. Riley, "Speech recognition with weighted finite-state transducers," in *Springer Handbook on Speech Processing and Speech Communication*, 2008, pp. 1–31.

[21] H. Bourlard and N. Morgan, *Connectionist speech recognition: a hybrid approach*. Kluwer Academic Publishers, 1994, vol. 247, no. 4.

[22] D. Povey, M. Hannemann, G. Boulianne, L. Burget, A. Ghoshal, M. Janda, M. Karafiát, S. Kombrink, P. Motlicek, Y. Qian *et al.*, "Generating exact lattices in the wfst framework," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4213–4216.

[23] J. G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (rover)," in *Automatic Speech Recognition and Understanding, 1997. Proceedings., 1997 IEEE Workshop on*. IEEE, 1997, pp. 347–354.

[24] H. Xu, D. Povey, L. Mangu, and J. Zhu, "Minimum bayes risk decoding and system combination based on a recursion for edit distance," *Computer Speech & Language*, vol. 25, no. 4, pp. 802–828, 2011.

[25] "Jeremy bentham, the works of jeremy bentham, 11 vols." [Online]. Available: http://oll.libertyfund.org/titles/bentham-works-of-jeremy-bentham-11-vols