A Neural Network based Approach to Automatic Post-Editing

Santanu Pal[1], Sudip Kumar Naskar[3], Mihaela Vela[1], Josef van Genabith[1,2]
[1]Saarland University, Saarbrücken, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Germany
[3]Jadavpur University, Kolkata, India
{santanu.pal, josef.vangenabith}@uni-saarland.de
sudip.naskar@cse.jdvu.ac.in, m.vela@mx.uni-saarland.de

Abstract

We present a neural network based automatic post-editing (APE) system to improve raw machine translation (MT) output. Our neural model of APE (NNAPE) is based on a bidirectional recurrent neural network (RNN) model and consists of an encoder that encodes an MT output into a fixed-length vector from which a decoder provides a post-edited (PE) translation. APE translations produced by NNAPE show statistically significant improvements of 3.96, 2.68 and 1.35 BLEU points absolute over the original MT, phrase-based APE and hierarchical APE outputs, respectively. Furthermore, human evaluation shows that the NNAPE generated PE translations are much better than the original MT output.

1 Introduction

For many applications the performance of state-of-the-art MT systems is useful but often far from perfect. MT technologies have gained wide acceptance in the localization industry. Computer aided translation (CAT) has become the de-facto standard in large parts of the translation industry which has resulted in a surge of demand for professional post-editors. This, in turn, has resulted in substantial quantities of PE data which can be used to develop APE systems.

In the context of MT, "post-editing" (PE) is defined as the correction performed by humans over the translations produced by an MT system (Veale and Way, 1997), often with minimal amount of manual effort (TAUS Report, 2010) and as a process of modification rather than revision (Loffler-Laurian, 1985).

MT systems primarily make two types of errors – lexical and reordering errors. However, due to the statistical and probabilistic nature of modelling in statistical MT (SMT), the currently dominant MT technology, it is non-trivial to rectify these errors in the SMT models. Post-edited data are often used in incremental MT frameworks as additional training material. However, often this does not fully exploit the potential of these rich PE data: e.g., PE data may just be drowned out by a large SMT model. An APE system trained on human post-edited data can serve as a MT post-processing module which can improve overall performance. An APE system can be considered as an MT system, translating predictable error patterns in MT output into their corresponding corrections.

APE systems assume the availability of source language input text (SL_{IP}), target language MT output (TL_{MT}) and target language PE data (TL_{PE}). An APE system can be modelled as an MT system between SL_{IP}-TL_{MT} and TL_{PE}. However, if we do not have access to SL_{IP}, but have sufficiently large amounts of parallel TL_{MT}-TL_{PE} data, we can still build an APE model between TL_{MT} and TL_{PE}.

Translations provided by state-of-the-art MT systems suffer from a number of errors including incorrect lexical choice, word ordering, word insertion, word deletion, etc. The APE work presented in this paper is an effort to improve the MT output by rectifying some of these errors. For this purpose we use a deep neural network (DNN) based approach. Neural MT (NMT) (Kalchbrenner and Blunsom, 2013; Cho et al., 2014a; Cho et al., 2014b) is a newly emerging approach to MT. On the one hand DNNs represent language in a continuous vector space which eases the modelling of semantic similarities (or distance) between phrases or sentences, and on the other hand it can also consider contextual information, e.g.,

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 281–286,
Berlin, Germany, August 7-12, 2016.

utilizing all available history information in deciding the next target word, which is not an easy task to model with standard APE systems.

Unlike phrase-based APE systems (Simard et al., 2007a; Simard et al., 2007b; Pal, 2015; Pal et al., 2015), our NNAPE system builds and trains a single, large neural network that accepts a 'draft' translation (TL_{MT}) and outputs an improved translation (TL_{PE}).

The remainder of the paper is organized as follows. Section 2 gives an overview of relevant related work. The proposed NNAPE system is described in detail in Section 3. We present the experimental setup in Section 4. Section 5 presents the results of automatic and human evaluation together with some analysis. Section 6 concludes the paper and provides avenues for future work.

2 Related Work

APE has proved to be an effective remedy to some of the inaccuracies in raw MT output. APE approaches cover a wide methodological range. Simard et al. (2007a) and Simard et al. (2007b) applied SMT for post-editing, handling the repetitive nature of errors typically made by rule-based MT systems. Lagarda et al. (2009) used statistical information from the trained SMT models for post-editing of rule-based MT output. Rosa et al. (2012) and Mareček et al. (2011) applied a rule-based approach to APE on the morphological level. Denkowski (2015) developed a method for real time integration of post-edited MT output into the translation model by extracting a grammar for each input sentence. Recent studies have even shown that the quality of MT plus PE can exceed the quality of human translation (Fiederer and OBrien, 2009; Koehn, 2009; DePalma and Kelly, 2009) as well as the productivity (Zampieri and Vela, 2014) in some cases.

Recently, a number of papers have presented the application of neural networks in MT (Kalchbrenner and Blunsom, 2013; **?**; Cho et al., 2014b; Bahdanau et al., 2014). These approaches typically consist of two components: an **encoder** encodes a source sentence and a **decoder** decodes into a target sentence.

In this paper we present a neural network based approach to automatic PE (NNAPE). Our NNAPE model is inspired by the MT work of Bahdanau et al. (2014) which is based on bidirectional recurrent neural networks (RNN). Unlike Bahdanau et al. (2014), we use LSTMs rather than GRUs as hidden units. RNNs allow processing of arbitrary length sequences, however, they are susceptible to the problem of vanishing and exploding gradients (Bengio et al., 1994). To tackle vanishing gradients in RNNs, two architectures are generally used: gated recurrent units (GRU) (Cho et al., 2014b) and long-short term memory (LSTM) (Hochreiter and Schmidhuber, 1997). According to empirical studies (Chung et al., 2014; Józefowicz et al., 2015) both architectures yield comparable performance. GRUs tend to train faster than LSTMs. On the other hand, given sufficient amounts of training data, LSTMs may lead to better results. Since our task is monolingual and we have more than 200K sentence pairs for training, we use a full LSTM (as the hidden units) to model our NNAPE system.

The model takes TL_{MT} as input and provides TL_{PE} as output. To the best of our knowledge the work presented in this paper is the first approach to APE using neural networks.

3 Neural Network based APE

The NNAPE system is based on a bidirectional (forward-backward) RNN based encoder-decoder.

3.1 A Bidirectional RNN APE Encoder-Decoder

Our NNAPE model encodes a variable-length sequence of TL_{MT} (e.g. $\mathbf{x} = x_1, x_2, x_3...x_m$) into a fixed-length vector representation and then decodes a given fixed-length vector representation back into a variable-length sequence of TL_{PE} (e.g. $\mathbf{y} = y_1, y_2, y_3...y_n$). Input and output sequence lengths, m and n, may differ.

A Bidirectional RNN encoder consists of forward and backward RNNs. The forward RNN encoder reads in each $\mathbf{x}$ sequentially from x_1 to x_m and at each time step t, the hidden state h_t of the RNN is updated by using a non-linear activation function f (Equation 1), an elementwise logistic sigmoid with an LSTM unit.

$$h_t = f(h_{t-1}, x_t) \quad (1)$$

Similarly, the backward RNN encoder reads the input sequence and calculates hidden states in reverse direction (i.e. x_m to x_1 and h_m to h_1 respectively). After reading the entire input sequence, the hidden state of the RNN is provided a summary c context vector ('**C**' in Figure 1) of the whole input sequence.

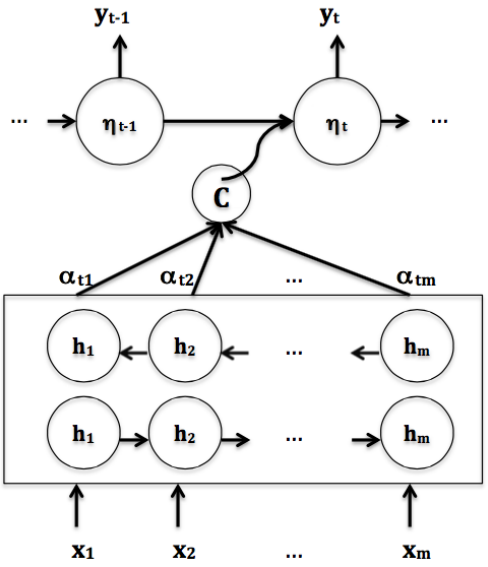

Figure 1: Generating the t^{th} TL_{PE} word y_t for a given TL_{MT} (**x**) by our NNAPE System.

The decoder is another RNN trained to generate the output sequence by predicting the next word y_t given the hidden state η_t and the context vector c_t (c.f., Figure1). The hidden state of the decoder at time t is computed as given below.

$$P(y_t|y_1, ...y_{t-1}, \mathbf{x}) = f(\eta_t, y_{t-1}, c_t) \quad (2)$$

$$\eta_t = f(\eta_{t-1}, y_{t-1}, c_t) \quad (3)$$

The context vector c_t can be computed as

$$c_t = \sum_{i=1}^{m} \alpha_{ti} h_i \quad (4)$$

Here, α_{ti}, is the weight of each h_i and can be computed as

$$\alpha_{ti} = \frac{exp(e_{ti})}{\sum_{j=1}^{m} exp(e_{tj})} \quad (5)$$

where $e_{ti} = a(\eta_{t-1}, h_i)$ is an alignment model which provides a matching score between the inputs around position *i* and the output at position *t*. The alignment score is based on the i^{th} annotation h_i of the input sentence and the RNN hidden state η_{t-1}. The alignment model itself is a feed-forward neural network which directly computes a soft alignment that allows the gradient of the cost function to be backpropagated through. The gradient is used to train the alignment model as well as the TL_{MT}–TL_{PE} translation model jointly.

The alignment model is computed $m \times n$ times as follows:

$$a(\eta_{t-1}, h_i) = v_a^T \tanh(W_a \eta_{t-1} + U_a h_i) \quad (6)$$

where $W_a \in \mathbf{R}^{n_h \times n_h}$, $U_a \in \mathbf{R}^{n_h \times 2n_h}$ and $v_a \in \mathbf{R}^{n_h}$ are the weight matrices of n_h hidden units.

4 Experiments

We evaluate the model on an English–Italian APE task, which is detailed in the following subsections.

4.1 Data

The training data used for the experiments was developed in the MateCat[1] project and consists of 312K TL_{MT}–TL_{PE} parallel sentences. The parallel sentences are (English to) Italian MT output and their corresponding (human) post-edited Italian translations. Google Translate (GT) is the MT engine which provided the original Italian TL_{MT} output. The data includes sentences from the Europarl corpus as well as news commentaries. Since the data contains some non-Italian sentences, we applied automatic language identification (Shuyo, 2010) in order to select only Italian sentences. Automatic cleaning and pre-processing of the data was carried out by sorting the entire parallel training corpus based on sentence length, filtering the parallel data on maximum allowable sentence length of 80 and sentence length ratio of 1:2 (either direction), removing duplicates and applying tokenization and punctuation normalization using Moses (Koehn et al., 2007) scripts. After cleaning the corpus we obtained a sentence-aligned TL_{MT}–TL_{PE} parallel corpus containing 213,795 sentence pairs. We randomly extracted 1000 sentence pairs each for the development set and test set from the pre-processed parallel corpus and used the remaining (211,795) as the training corpus. The training data features 57,568 and 61,582 unique word types in TL_{MT} and TL_{PE}, respectively. We chose the 40,000 most frequent words from both TL_{MT} and TL_{PE} to train our NNAPE model. The remaining words which are not among the most frequent words are replaced by a special token ([UNK]). The model was trained for approximately 35 days, which is equivalent to 2,000,000 updates with GPU settings.

4.2 Experimental Settings

Our bidirectional RNN Encoder-Decoder contains 1000 hidden units for the forward backward RNN encoder and 1000 hidden units for the decoder.

[1]https://www.matecat.com/

The network is basically a multilateral neural network with a single maxout unit as hidden layer (Goodfellow et al., 2013) to compute the conditional probability of each target word. The word embedding vector dimension is 620 and the size of the maxout hidden layer in the deep output is 500. The number of hidden units in the alignment model is 1000. The model has been trained on a mini-batched stochastic gradient descent (SGD) with 'Adadelta' (Zeiler, 2012). The main reason behind the use of 'Adadelta' is to automatically adapt the learning rate of each parameter ($\epsilon = 10^{-6}$ and $\rho = 0.95$). Each SGD update direction is computed using a mini-batch of 80 sentences.

We compare our NNAPE system with state-of-the-art phrase-based (Simard et al., 2007b) as well as hierarchical phrase-based APE (Pal, 2015) systems. We also compare the output provided by our system against the original GT output. For building the phrase-based and hierarchical phrase-based APE systems, we set maximum phrase length to 7. A 5-gram language model built using KenLM (Heafield, 2011) was used for decoding. System tuning was carried out using both k-best MIRA (Cherry and Foster, 2012) and Minimum Error Rate Training (MERT) (Och, 2003) on the held-out development set (devset). After parameters were tuned, decoding was carried out on the held out test set.

5 Evaluation

The performance of the NNAPE system was evaluated using both automatic and human evaluation methods, as described below.

5.1 Automatic Evaluation

The output of the NNAPE system on the 1000 sentences testset was evaluated using three MT evaluation metrics: BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and Meteor (Denkowski and Lavie, 2011). Table 1 provides a comparison of our neural system performance against the baseline phrase-based APE (S_1), baseline hierarchical phrase-based APE (S_2) and the original GT output. We use *a*, *b*, *c*, and *d* to indicate statistical significance over GT, S_1, S_2 and our NNAPE system (NN), respectively. For example, the S_2 BLEU score $63.87_{a,b}$ in Table 1 means that the improvement provided by S_2 in BLEU is statistically significant over Google Translator and phrase-based APE. Table 1 shows that S_1 provides statistically significant ($0.01 < p < 0.04$) improvements over GT across all metrics. Similarly S_2 yields statistically significant ($p < 0.01$) improvements over both GT and S_1 across all metrics. The NN system performs best and results in statistically significant ($p < 0.01$) improvements over all other systems across all metrics. A systematic trend ($NN > S_2 > S_1 > GT$) can be observed in Table 1 and the improvements are consistent across the different metrics. The relative performance gain achieved by NN over GT is highest in TER.

System	BLEU	TER	METEOR
GT (a)	61.26	30.94	72.73
S_1 (b)	62.54_{a}	29.49_{a}	73.21_{a}
S_2 (c)	$63.87_{a,b}$	$28.67_{a,b}$	$73.63_{a,b}$
NN (d)	$\mathbf{65.22}_{a,b,c}$	$\mathbf{27.56}_{a,b,c}$	$\mathbf{74.59}_{a,b,c}$

Table 1: Automatic evaluation.

5.2 Human Evaluation

Human evaluation was carried out by four professional translators, native speakers of Italian, with professional translation experience between one and two years. Since human evaluation is very costly and time consuming, it was carried out on a small portion of the test set consisting of 145 randomly sampled sentences and only compared NN with the original GT output. We used a polling scheme with three different options. Translators choose which of the two (GT or NN) outputs is the better translation or whether there is a tie ('uncertain'). To avoid any bias towards any particular system, the order in which two system outputs are presented is randomized so that the translators do not know which system they are contributing their votes to.

We analyzed the outcome of the voting process (4 translators each giving 145 votes) and found that the winning NN system received 285 (49.13%) votes compared to 99 (17.07%) votes received by the GT system, while the rest of the votes (196, 33.79%) go to the 'uncertain' option. We measured pairwise inter-annotator agreement between the translators by computing Cohen's κ coefficient (Cohen, 1960) reported in Table 2. The overall κ coefficient is 0.330. According to (Landis and Koch, 1977) this correlation coefficient can be interpreted as fair.

Cohen's κ	T1	T2	T3	T4
T1	-	0.141	0.424	0.398
T2	0.141	-	0.232	0.540
T3	0.424	0.232	-	0.248
T4	0.398	0.540	0.248	-

Table 2: Pairwise correlation between translators in the evaluation process.

5.3 Analysis

The results of the automatic evaluation show that NNAPE has advantages over the phrase-based and hierarchical APE approaches. On manual inspection we found that the NNAPE system drastically reduced the preposition insertion and deletion error in Italian GT output and was also able to handle the improper use of prepositions and determiners (e.g. "states" → "dei stati", "the states" → "gli stati"). The use of a bidirectional RNN neural model makes the model sensitive towards contexts. Moreover, NNAPE captures global reordering by capturing contextual features which helps to reduce word ordering errors to some extent.

6 Conclusion and Future Work

The NNAPE system provides statistically significant improvements over existing state-of-the-art APE models and produces significantly better translations than GT which is very difficult to beat. This enhancement in translation quality through APE should reduce human PE effort. Human evaluation revealed that the NNAPE generated PE translations contain less lexical errors, NNAPE rectifies erroneous word insertions and deletions, and improves word ordering.

In future, we would like to test our system in a real-life translation scenario to analyze productivity gains in a commercial environment. We also want to extend the APE system by incorporating source language knowledge into the network and compare LSTM against GRU hidden units.

Acknowledgments

We would like to thank all the anonymous reviewers for their feedback. We are also thankful to Translated SRL, Rome, Italy. They have shared their data for the experiments and enabled the manual evaluation of our system.

Santanu Pal is supported by the People Programme (Marie Curie Actions) of the European Union's Framework Programme (FP7/2007-2013) under REA grant agreement no 317471.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning Long-Term Dependencies with Gradient Descent is Difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436.

KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *CoRR*, abs/1409.1259.

Kyunghyun Cho, Bart Van Merriënboer, Çalar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

Junyoung Chung, Çalar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. Technical Report Arxiv report 1412.3555, Université de Montréal.

Jacob Cohen. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, 20(1):37–46, April.

Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*, pages 85–91.

Michael Denkowski. 2015. *Machine Translation for Human Translators*. Ph.D. thesis, Carnegie Mellon University.

Donald A. DePalma and Nataly Kelly. 2009. Project Management for Crowdsourced Translation: How User-Translated Content Projects Work in Real Life. *Translation and Localization Project Management: The Art of the Possible*, pages 379–408.

Rebecca Fiederer and Sharon OBrien. 2009. Quality and Machine Translation: a Realistic Objective. *Journal of Specialised Translation*, 11:52–74.

Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. *arXiv preprint arXiv:1302.4389*.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780, November.

Rafal Józefowicz, Wojciech Zaremba, and Ilya Sutskever. 2015. An Empirical Exploration of Recurrent Network Architectures. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2342–2350.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent Continuous Translation Models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180.

Philipp Koehn. 2009. A Process Study of Computer-aided Translation. *Machine Translation*, 23(4):241–263.

Antonio Lagarda, Vicent Alabau, Francisco Casacuberta, Roberto Silva, and Enrique Díaz-de Liaño. 2009. Statistical post-editing of a rule-based machine translation system. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 217–220.

J. Richard Landis and Gary G. Koch. 1977. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–74.

Anne-Marie Loffler-Laurian. 1985. Traduction Automatique et Style. *Babel*, 31(2):70–76.

David Mareček, Rudolf Rosa, Petra Galuščáková, and Ondřej Bojar. 2011. Two-step Translation with Grammatical Post-processing. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 426–432.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 160–167.

Santanu Pal, Mihaela Vela, Sudip Kumar Naskar, and Josef van Genabith. 2015. USAAR-SAPE: An English–Spanish Statistical Automatic Post-Editing System. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 216–221.

Santanu Pal. 2015. Statistical Automatic Post Editing. In *The Proceedings of the EXPERT Scientific and Technological workshop*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318.

Rudolf Rosa, David Mareček, and Ondřej Dušek. 2012. DEPFIX: A System for Automatic Correction of Czech MT Outputs. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 362–368.

Nakatani Shuyo. 2010. Language Detection Library for Java.

Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007a. Statistical Phrase-Based Post-Editing. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 508–515.

Michel Simard, Nicola Ueffing, Pierre Isabelle, and Roland Kuhn. 2007b. Rule-Based Translation with Statistical Phrase-Based Post-Editing. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 203–206.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of association for machine translation in the Americas*, pages 223–231.

TAUS Report. 2010. Post editing in practice. Technical report, TAUS.

Tony Veale and Andy Way. 1997. Gaijin: A Bootstrapping, Template-driven Approach to Example-based MT. In *Proceedings of the Recent Advances in Natural Language Processing*.

Marcos Zampieri and Mihaela Vela. 2014. Quantifying the Influence of MT Output in the Translators Performance: A Case Study in Technical Translation. In *Proceedings of the EACL Workshop on Humans and Computer-assisted Translation (HaCat)*, pages 93–98.

Matthew D. Zeiler. 2012. ADADELTA: An Adaptive Learning Rate Method. *arXiv:1212.5701 [cs.LG]*.