R222 Assignment

Kaho Sato
King's College
ks789@cam.ac.uk

1 Introduction

There is an increasing interest in *native language identification* (NLI). NLI is a task of detecting the native language (L1) of the author of the text. This may be treated as a sub-task of *author profiling*, where the goal is to predict traits of the author such as gender, age, or psychometric traits (Estival et al., 2007). Author profiling is used, for instance, to narrow down a list of suspects of criminal activities (Abbasi and Chen, 2005) by filtering it with the traits detected from the communication on online mediums. A possible commercial application of author profiling is collection of customer information from the reviews for market intelligence (Glance et al., 2005), which could be especially effective when used in conjunction with sentiment analysis.

In addition to these applications of author profiling, NLI can be used to create a better writing tutor system for a non-native speaker of English. When using a non-native language (L2), one often applies the knowledge of their L1. This well-studied phenomenon is called *L1 transfer* (Wanner and Gleitman, 1982; Frenck-Mestre and Pynte, 1997; Dussias, 2003; Nitschke et al., 2010). L1 transfer could result to correct use of the language, but could also lead to errors which are specific to the L1. There are several works which are based on this observation and investigate how the knowledge of the authors' L1 could improve the grammatical error detection and correction in their texts (Chang et al., 2008; Rozovskaya and Roth, 2010, 2011; Dahlmeier and Ng, 2011). A writing tutor system can first use NLI, and use the detected L1 to have an improved error detection and correction.

NLI has been extensively studied in the past decade. Early works in NLI (Koppel et al., 2005; Tsur and Rappoport, 2007) showed the effectiveness of Support Vector Machine (SVM) in this task. SVM continues to be the popular choice, and 13 out of 24 participating teams, including the winning team (Jarvis et al., 2013) in the First Native Language Identification Shared Task (Tetreault et al., 2013) utilised SVM. The difference among these works comes down to the feature sets used, which range from lexical n-grams (Koppel et al., 2005; Tsur and Rappoport, 2007; Jarvis et al., 2013) to fragments of Tree Substitutional Grammar (Swanson and Charniak, 2012). In other words, feature engineering has been a large focus in the field of NLI.

In this report, we perform NLI on a subset of Cambridge Learner Corpus (Nicholls, 2003; Yannakoudakis et al., 2011) using a character-based convolutional neural network (CNN). Specifically, we use the character-based CNN designed to perform various classification tasks, presented by Zhang et al. (2015). We chose to use a character-based CNN as opposed to a word-based CNN, as lexical features are in general considered to be more appropriate for the task of topic categorisation than of author profiling (Kochmar, 2011). CNNs have gained in popularity for its success in various practical applications, such as image recognition (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014; He et al., 2016), natural language processing (Jackson and Moulinier, 2007; Collobert et al., 2011b; Kalchbrenner et al., 2014), and playing Go (Silver et al., 2016).

CNNs have a great advantage over traditional machine learning techniques such as SVM, which is the lack of feature engineering. The downside is that they typically require a large-scale dataset, and the dataset we have is fairly small. Indeed, applying the implementation by Zhang et al. (2015) with no modification to the dataset does not perform better than a simple baseline system which assigns the label arbitrarily. Another potential explanation for the poor performance is the length of each document in the dataset. Zhang et al. (2015)

uses the first 1014 characters of each document as an input. However, many of the documents are shorter than 1014 characters and therefore require padding. This may have introduced some noise to the input.

To partially mitigate these problems, we use a smaller window and sample a section of the document at an arbitrary location, rather than always taking the first part. The results show that CNN performs better with a small window size. However, the performance still does not match that of SVM.

The remaining of the report proceeds as follows; Section 2 briefly discusses the previous works in NLI; Section 3 gives the convolution used in the CNN we experimented with; Section 4 discusses the dataset we perform NLI on; Section 5 describes the models we experimented with; Section 6 presents and discusses the results; Finally, Section 7 gives a brief summary of the report.

2 Related Work

The first work on NLI that we are aware of is that of Koppel et al. (2005). They used SVM to perform five-class classification between Russian, Czech, Bulgarian, French, and Spanish. The features they extracted include stylistic features such as function words, character n-grams, and grammatical errors. They report the accuracy of 80.2%. Tsur and Rappoport (2007) performs the exact same task using SVM and investigates the effect of each features used. Wong and Dras (2011) experiment with substructures of a parse tree as additional features and achieve the accuracy of 80% in seven-class classification with SVM. Swanson and Charniak (2012) also used SVM and fragments of Tree Substitutional Grammar as additional features. This gave 78.4% accuracy in the same task as Wong and Dras (2011). Swanson and Charniak (2012) replicate the work by Wong and Dras (2011) and report the accuracy of 72.6%.

Aside from SVM, Latent Dirichlet Analysis was applied to NLI with seven languages by Dras and Johnson (2011). This gave the accuracy of 56.9 %, which was lower than 64.1% accuracy that their baseline SVM gave. All the works mentioned up to this point uses the ICLE corpus (Granger et al., 2002).

In 2013, the First Native Language Identification Shared Task was held (Tetreault et al., 2013). The task was to perform NLI on the new TOEFL11 corpus (Tetreault et al., 2013) which included essays written by native speakers of 11 languages.

As mentioned previously, the majority, including the winning team (Jarvis et al., 2013), utilised SVM. Jarvis et al. (2013) achieves the accuracy of 83.6%.

The other approaches seen in the shared task include MaxEnt, Ensemble, and Discriminant Function Analysis (Tetreault et al., 2013).

3 Background

In this section, we describe one dimensional CNNs which are commonly used in natural language processing(Collobert and Weston, 2008; Collobert et al., 2011b; Kalchbrenner et al., 2014; Zhang et al., 2015; Goldberg, 2016).

CNNs are made up of two parts; convolutional layers and fully-connected layers. The role of convolutional layers is to produce the most salient information of the input (Goldberg, 2016). The output of the convolutional layers are then fed to the fully-connected layers, which perform the prediction. We omit the description of fully-connected layers.

A convolutional layer applies three operations to its input. The first operation is *convolution*, which is the application of a linear function to a window of the input. The window is moved with some stride, and the convolution is applied to each window (Goldberg, 2016). Let l be the number of the features which represent the input, and k be the width of the sliding window. Let $f(x)$ be a function which takes an index in the window $x \in [1, k]$ and returns the weight applied to the xth element. Let $g(x)$ be a function which takes an index of the input $x \in [1, l]$ and returns the xth feature in the input. Let d be the stride of the convolution. Then, the result of the convolution applied to yth window is expressed as the following function $h(y)$ (Zhang et al., 2015):

$$h(y) = \sum_{x=1}^{k} f(x) \cdot g(y \cdot d - x + c)$$

where $c = k - d + 1$ is an offset constant.

The second operation is *non-linearlity* which is applied to each result of the convolution. This allows CNNs to model a non-linear function. A pop-

ular choice of non-linearlity is *rectifier*, defined as:

$$ReLU(x) = max(0, x)$$

The last operation in a convolutional layer is *pooling*. Similarly to convolution, the pooling operation is also done with a sliding window, moved with a certain stride. In pooling, the information in the window is aggregated by some function. For instance, in max-pooling, the function returns the largest element in the window is used. Another popular function for pooling returns the average of the elements in the window. The purpose of pooling is to reduce the to possibility of overfitting by downsizing the input and reducing the number of parameters in the later layers (Krizhevsky et al., 2012).

In the training, the loss is calculated from the output of the fully-connected network, and the error gradients are propagated back to the convolutional layers to adjust the weight applied by $f(x)$ in convolution (Goldberg, 2016). Thus the features that the convolutional layers extract from the input are most informative for the particular prediction task the network is trained for.

4 Dataset

We use a subset of the Cambridge Learner Corpus (Nicholls, 2003) (CLC). CLC is a corpus which consists of scripts produced by learners with 86 different L1 for various Cambridge English Language Assessment examinations. Each document is annotated with basic information of the author, such as their L1 and age, making CLC appropriate corpus for author profiling.

In particular, we use a subset of the publicly available CLC FCE Dataset (Yannakoudakis et al., 2011) as a test set. This contains 1244 scripts, each containing two answers, produced by learners with 16 different L1 for the Cambridge ESOL First Certificate in English (FCE) examination in 2010 and 2011.

Table 1 shows the number of scripts for each L1 in the CLC FCE Dataset. We extracted scripts produced by Japanese, Russian, and Italian native speakers to create a test set of 476 examples (Recall that each script contains two answers). We chose these three languages for two reasons; Firstly, the number of the scripts available for each language is fairly even. Secondly, they belong to the separate language families, which should make the classification more manageable. We selected 3000 answers from the CLC, 1000 for each L1, to build a training set.

Table 1: Distribution of L1 in the CLC FCE Dataset

L1	Count
Dutch	2
Swedish	15
Thai	63
Catalan	64
Chinese	66
Portuguese	68
German	69
Greek	74
Italian	76
Polish	76
Turkish	77
Japanese	80
Russian	82
Korean	86
French	146
Spanish	200

The level of proficiency is another variable that influences the writing of a learner, and this may introduce an undesirable bias. Each Cambridge ESOL examination expects a certain level of proficiency from the candidates. The reference levels are provided by Common European Framework of Reference for Languages (CEFR) (COUNCIL, 2001), and FCE is aimed for learners at CEFR level B (i.e. independent user). For the training set, for each language, we selected 250 answers by learners at CEFR level A and C (i.e. basic and proficient user, respectively) and 500 answers by learners at CEFR level B. We included more answers produced by learners at CEFR level B in order to assimilate the training set to the test set. Besides, it is reasonable to assume that there are more independent users than basic or proficient users.

Table 2 and 3 gives the statistic of the character counts of the documents in the training and test set respectively. As can be seen, there is a much larger variance in the training set than in the test set. This is due to the fact that the training set contains answers for 15 different exams, where the expected length of the answers is different from one another, whereas the test set only contains answers for FCE.

From the training set, we took 300 examples to create a validation set. The error rate on the valida-

Table 2: Statistics of the Character Length of Documents in Training Set

Min	58
Max	3710
Median	376.5
Average	742.1
Standard Deviation	716.4

Table 3: Statistics of the Character Length of Documents in Test Set

Min	685
Max	1822
Median	1081
Average	1099
Standard Deviation	186.8

tion set is computed after each epoch and was used to see the length of training which does not make the model to underfit or overfit to the remaining 2700 examples.

5 Models

This section describes the models we experimented with. First, we describe baseline systems. Then we outline the CNN described in (Zhang et al., 2015) and the modifications we made.

5.1 Baselines

We have two baseline systems to compare the CNN against; One is `majority` baseline, which assigns the label which occurs most frequently in the test set (i.e. Russian). The other is `simple SVM` baseline, which uses word unigram count as a feature. We do not apply any feature selection.

5.2 CNN

5.2.1 Input

We first need to encode documents to a fixed sized sequence of vectors to create an input for a convolutional layer described in Section 3. Zhang et al. (2015) do this by lowercasing the document, and mapping each character to a one-hot vector. The alphabet used in our experiment is the same as in (Zhang et al., 2015) and is

```
abcdefghijklmnopqrstuvwxyz
0123456789
-,;.!?:"/\|_@#$%&*+-=<>()[]{}
```

As the alphabet consists of 70 characters, each document is transformed into a sequence of vectors of size 70. If a character in a document is not present in the alphabet, it is encoded to an all-zero vector. Note that a whitespace is not part of the alphabet and therefore is also encoded to an all-zero vector.

Zhang et al. (2015) perform a data augmentation in order to avoid the generalisation error. Specifically, they use an English thesaurus which is based on WordNet (Fellbaum, 1998) and replace words in the document with their synonyms to create a "new" example. The number of words to be replaced and the synonym used is chosen probabilistically.

This data augmentation technique is appropriate if the task is to classify the documents according to its semantics. Indeed, the text classification tasks that Zhang et al. (2015) use this CNN for are sentiment analysis and topic classification. Though the dataset we have is small and a data augmentation is necessary, it was expected that this particular technique would introduce an undesirable noise in training for author profiling. This is because as word choice is a crucial clue to the attributes of the author, especially their L1. In fact, Yarowsky (2013) shows that the frequency distribution of words used in a corpus of computational linguistics articles from the ACL Anthology varies greatly from L1 to L1 of the author. For instance, a word "claim" is much less often used by Chinese native speakers than native speakers of other languages, and "complementary" is more frequently used by French or Spanish native speakers. The latter can potentially be explained by the latin origin of the word. To preserve such distinguishing features in the data, we turned off the data augmentation.

In (Zhang et al., 2015), the first 1014 characters are taken from each document and encoded to be used as an input. This is potentially problematic with the dataset we have. As shown in Table 2, majority of the documents in the training set is significantly shorter than 1014 characters. Though pooling should help the model to be invariant to the location of a feature, having majority of the training examples padded extensively might be harmful.

To partially mitigate the problem of small dataset and short documents, during the training, instead of feeding the first 1014 characters of the document to the model, we take a window of smaller size at an arbitrary location. For instance,

given a document of length 1000 and a window size $l_w = 100$, a CNN may sample 100 characters at 901 different locations, effectively creating 900 different examples. In this report, we refer to this approach as *random window sampling*. For testing, we take the window of the same size at the beginning of the document for an easier comparison of the results.

We experiment with $l_w \in \{123, 339, 555, 771, 1014\}$. Note that these numbers are chosen as they are compatible with the architecture of the CNN.

5.2.2 Architecture

Figure 1 gives an illustration of the model described in (Zhang et al., 2015). The CNN consists of six convolutional and three fully-connected layers. All the convolutional layers apply one dimensional convolution, which was described in Section 3. The non-linearlity used is a threshold function th defined as follows:

$$th(x) = \begin{cases} x & \text{if } x > 1\text{e}{-6} \\ 0 & \text{otherwise} \end{cases}$$

Max-pooling is applied with stride three at the first, second, and sixth layer. Table 4 shows the configuration of the convolutional layers. The number of features of the input is 70, which is the size of the alphabet given in 5.2.1. As stated previously, in (Zhang et al., 2015), the length of the input is 1014. With our approach of random window sampling, the length of the input is equal to the window size l_w. The output frame length after the last convolutional layer is $(l_0 - 96)/27$, where l_0 is the length of the input. This defines a set of possible window sizes l_w.

Between the fully-connected layers, dropout modules (Hinton et al., 2012) are used. Dropout modules prevent co-adaptation of hidden units by setting them to zero at a certain probability p. In this CNN, $p = 0.5$ is used. The seventh and eighth layer have 1024 output units, and ninth layer have three as this is applied to three-class classification.

At each epoch, a minibatch of size 128, chosen randomly from the training set, is used to update the weight with stochastic gradient decent (Polyak, 1964) with momentum (Sutskever et al., 2013) 0.9 and step size 0.01.

The original implementation of this CNN is written in Torch (Collobert et al., 2011a) and available online[1].

[1] `https://github.com/zhangxiangxiao/Crepe`

6 Results

Table 5 shows the accuracy of the baselines and the CNN with the input size 1014. Model (A) is the unmodified implementation described in (Zhang et al., 2015). As described in Section 5.2.1, we modify the implementation in two ways; one is applying the random window sampling, and the other is switching off the data augmentation using a thesaurus.

Model (A), (C), and (D) perform better than the `majority` baseline, though only by a small margin. `simple SVM` outperforms all the other systems.

Model (D) outperforms all the other CNN models, showing the effectiveness of the two modifications.

We argue that random window sampling is an effective data augmentation method for this task. As stated previously, this CNN is shown effective for sentiment analysis and topic categorisation (Zhang et al., 2015). In (Zhang et al., 2015), the types of documents treated are reviews or articles, and the first part of these documents tend to be more informative about the overall semantics of the documents which is the objective of the task. Using only the start of the document, therefore, means less noise in the training data for the problems discussed in (Zhang et al., 2015). However, in the context of NLI, there is no clear relation between the informativeness of a feature and its location. Therefore, it is useful to augment the small dataset with random window sampling, rather than discarding possibly useful information which may be contained in the later part of the training example.

A possible explanation of model (A) performing better than model (B) is that the data augmentation using a thesaurus helps counteract the bias introduced by using only the first part of the document. This bias may be related to, for instance, the topic of the document or to the format that the exam prompt asks for. There is an imbalance in the exams taken by native speakers of each language. This means that there may be a prompt on a particular topic or asks for a particular format which was chosen more frequently by learners of a certain L1. Clearly, if a network learns to classify based on the topic or the format of the document, it would not generalise and perform poorly on the test set. As mentioned previously, the first part of the document may contain more information about

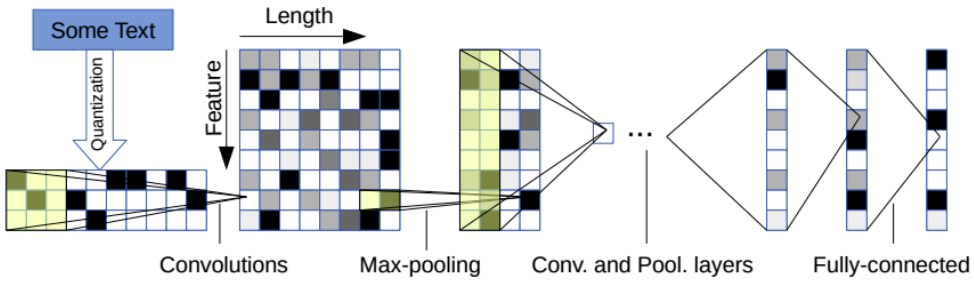

Figure 1: Overview of Architecture

Table 4: Convolutional Layers Used in (Zhang et al., 2015)

Layer	Frame Size	Kernel Width	Pooling Region Width
1	256	7	3
2	256	7	3
3	256	3	N/A
4	256	3	N/A
5	256	3	N/A
6	256	3	3

the topic, and it possibly is more affected by the format. For instance, there are many documents which are in the format of a letter. These always start with a greeting (e.g. Dear ...), and if there are more of such documents present for a certain L1, the network may associate the greeting with this L1.

Table 6 shows the accuracies of the model with different window sizes. Again, all the models except for model (G) perform better than the `majority` baseline system, though only by small margin. `simple SVM` outperforms all the other systems by the large margin.

Interestingly, the best performances were achieved with the smallest and the largest window size. This suggests that there is a trade-off between a small and a large window. Intuitively, having a large window would mean that the network is fed with more information, which is beneficial. It, however, also means that the network is fed with more padding. Table 7 shows the proportion of examples which are padded in each dataset. As can be seen, with a window size larger than 123, a large proportion of the training set is padded. Though the padding itself potentially introduces some noise, it affected the network less with the window size 1014, perhaps because also many examples in the test set were padded. As a future work, one may filter short examples which require padding from training set and see whether the performance improves.

Table 8 shows the confusion matrix for model (D). As can be seen, a large proportion of documents are labelled with Russian. This is perhaps due to features learnt to be characteristic of Russian speakers writing was something that occur often in English writing in general.

A possible explanation to a large difference between the performance of the CNN applied to this task and the tasks introduced in (Zhang et al., 2015) is the size of dataset. Zhang et al. (2015) states that convolutional networks, especially when learning from low-level raw features (i.e. characters) requires a large-scale dataset, and the smallest dataset Zhang et al. (2015) experiment with consists of 120k training examples with equal splits between four classes. This is significantly larger than our dataset. Stab and Gurevych (2017), however, uses a convolutional network trained only on 1029 examples and achieve the accuracy 0.843 ± 0.025 on the task of classifying arguments as sufficient or not. This is much higher than that of their majority baseline, which is 0.662 ± 0.033. This network takes word embeddings produced using word2vec (Mikolov et al., 2013). Though lexical features are not considered to be appropriate for the task of NLI as stated previously, as a future work it may be worth investigating the use of word-based CNN for NLI.

Table 5: Accuracy of Models with Input Size 1014

	Random Window Sampling	Augmentation	Accuracy
(A)	No	Yes	0.352
(B)	No	No	0.336
(C)	Yes	Yes	0.360
(D)	Yes	No	0.374
`majority`	N/A	N/A	0.345
`simple SVM`	N/A	N/A	0.613

Table 6: Accuracy of Models with Random Window Sampling

	Input Size	Accuracy
(D)	123	0.422
(E)	339	0.368
(F)	555	0.346
(G)	771	0.318
(C)	1014	0.374
`majority`	N/A	0.345
`simple SVM`	N/A	0.613

Table 7: Proportion of Examples with Padding

Input Size	Training	Test
123	1.20%	0%
339	46.5%	0%
555	58.5%	0%
771	68.7%	2.5%
1014	74.3%	33.6%

Table 8: Confusion Matrix of Model (D)

		predicted		
		IT	RU	JP
label	IT	43	74	35
	RU	22	95	47
	JP	18	79	63

7 Conclusion

In this report, we investigated whether a character-based CNN designed by Zhang et al. (2015) is effective for NLI. Though it did perform better than `majority` baseline system, the improvement was minuscule. We experimented with two modifications to their implementation. One is to switch off the data augmentation using thesaurus, and the other is to use random window sampling. Each modification alone gave no improvement. However, when both modifications were applied, the accuracy went up by 2.2% with input size 1014.

We also experimented with a various window sizes and found that the model which used the window of size 123 performed the best. This gave an improvement of 7.7% to the `majority` baseline and 7.0% to the original implementation by Zhang et al. (2015).

However, `simple SVM` outperforms all the CNN models. As a future work one may experiment with word-based CNN or the dataset where short documents are filtered out.

References

Ahmed Abbasi and Hsinchun Chen. Applying authorship analysis to arabic web content. In *International Conference on Intelligence and Security Informatics*, pages 183–197. Springer, 2005.

Yu-Chia Chang, Jason S Chang, Hao-Jan Chen, and Hsien-Chin Liou. An automatic collocation writing assistant for taiwanese efl learners: A case of corpus-based nlp technology. *Computer Assisted Language Learning*, 21(3):283–299, 2008.

R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011a.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM, 2008.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537, 2011b.

OF EUROPE COUNCIL. Common european framework of reference for lan-

guages: Learning, teaching, assessment (cefr). *Electronic version:¡ http://www. coe. int/t/dg4/linguistic/cadre1_en. asp*, 2001.

Daniel Dahlmeier and Hwee Tou Ng. Correcting semantic collocation errors with l1-induced paraphrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 107–117. Association for Computational Linguistics, 2011.

Sze-Meng Jojo Wong Mark Dras and Mark Johnson. Topic modeling for native language identification. In *Australasian Language Technology Association Workshop 2011*, page 115, 2011.

Paola E Dussias. Syntactic ambiguity resolution in l2 learners. *Studies in Second Language Acquisition*, 25(04):529–557, 2003.

Dominique Estival, Tanja Gaustad, Son Bao Pham, Will Radford, and Ben Hutchinson. Author profiling for english emails. In *Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics (PACLING07)*, pages 263–272, 2007.

Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

Cheryl Frenck-Mestre and Joel Pynte. Syntactic ambiguity resolution while reading in second and native languages. *The Quarterly Journal of Experimental Psychology A*, 50(1):119–148, 1997.

Natalie Glance, Matthew Hurst, Kamal Nigam, Matthew Siegler, Robert Stockton, and Takashi Tomokiyo. Deriving marketing intelligence from online discussion. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 419–428. ACM, 2005.

Yoav Goldberg. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420, 2016.

Sylviane Granger, Estelle Dagneaux, Fanny Meunier, and Magali Paquot. *International corpus of learner English*. Presses universitaires de Louvain, 2002.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Peter Jackson and Isabelle Moulinier. *Natural language processing for online applications: Text retrieval, extraction and categorization*, volume 5. John Benjamins Publishing, 2007.

Scott Jarvis, Yves Bestgen, and Steve Pepper. Maximizing classification accuracy in native language identification. 2013.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Ekaterina Kochmar. *Identification of a writers native language by error analysis*. PhD thesis, 2011.

Moshe Koppel, Jonathan Schler, and Kfir Zigdon. Determining an author's native language by mining a text for errors. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 624–628. ACM, 2005.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

Diane Nicholls. The cambridge learner corpus: Error coding and analysis for lexicography and elt. 2003.

Sanjo Nitschke, Evan Kidd, and Ludovica Serratrice. First language transfer and long-term structural priming in comprehension. *Language and Cognitive Processes*, 25(1):94–114, 2010.

Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

Alla Rozovskaya and Dan Roth. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 conference on*

empirical methods in natural language processing, pages 961–970. Association for Computational Linguistics, 2010.

Alla Rozovskaya and Dan Roth. Algorithm selection and model adaptation for esl correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 924–933. Association for Computational Linguistics, 2011.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Stab and Iryna Gurevych. Recognizing insufficiently supported arguments in argumentative essays. 2017.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147, 2013.

Ben Swanson and Eugene Charniak. Native language detection with tree substitution grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 193–197. Association for Computational Linguistics, 2012.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. A report on the first native language identification shared task. 2013.

Oren Tsur and Ari Rappoport. Using classifier features for studying the effect of native language on the choice of written second language words. In *Proceedings of the Workshop on Cognitive Aspects of Computational Language Acquisition*, pages 9–16. Association for Computational Linguistics, 2007.

Eric Wanner and Lila R Gleitman. *Language acquisition: The state of the art*. CUP Archive, 1982.

Sze-Meng Jojo Wong and Mark Dras. Exploiting parse structures for native language identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610. Association for Computational Linguistics, 2011.

Sze-Meng Jojo Wong, Mark Dras, and Mark Johnson. Exploring adaptor grammars for native language identification. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 699–709. Association for Computational Linguistics, 2012.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 180–189. Association for Computational Linguistics, 2011.

Shane BergsmaDavid Yarowsky. Learning domain-specific, l1-specific measures of word readability. 2013.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.