Siamese Convolutional Networks for Cognate Identification

Taraka Rama
Department of Linguistics
University of Tübingen, Germany
taraka-rama.kasicheyanula@uni-tuebingen.de

Abstract

In this paper, we present phoneme level Siamese convolutional networks for the task of pair-wise cognate identification. We represent a word as a two-dimensional matrix and employ a siamese convolutional network for learning deep representations. We present siamese architectures that jointly learn phoneme level feature representations and language relatedness from raw words for cognate identification. Compared to previous works, we train and test on larger and realistic datasets; and, show that siamese architectures consistently perform better than traditional linear classifier approach.

1 Introduction

Cognates are words that are known to have descended from a common ancestral language. In historical linguistics, identification of cognates is an important step for positing relationships between languages. An example of cognates are German *Fuß* and English *foot* whereas, Hindi *chakra* and English *wheel* are cognates that can be traced back to the Proto-Indo-European $*k^wek^wlo-$ and do not exhibit similarity on surface.

In NLP, automatic identification of cognates is associated with the task of determining if two words are descended from a common ancestor or not. In NLP, word similarity measures based on number of shared bi-grams, minimum-edit-distance, and length of longest common subsequence are supplied as features for a linear classifier or a sequence labeler on a set of labeled positive and negative examples; and then employ the trained classifier to classify new word pairs. The features for a classifier consist of string similarity scores (Hauer and Kondrak, 2011; Inkpen et al., 2005).

It has to be noted that the Indo-European dating studies (Bouckaert et al., 2012; Chang et al., 2015; Rama, 2016) employ human expert cognacy judgments for inferring phylogeny and internal dates of a well-studied language family. Therefore, there is a need for developing automated cognate identification methods that can be applied to those families of the world that are not as well-studied as Indo-European language family.

The supervised approaches (Kondrak, 2009; Bergsma and Kondrak, 2007) employ orthographic similarities and character alignments as features for training classifiers. In this work, we show how convolutional networks can be employed to extract phonetic features for the purpose of cognate identification. We also include a neural network approach to integrate language features for jointly training the neural networks. To the best of our knowledge, this work is the first to apply convolutional networks (CNN) for the purpose of cognate identification.

The work is organized as follows. In section 2, we define the task of cognate identification. In section 3, we motivate and describe convolutional network architectures for cognate identification. In section 4, we describe the related work for cognate identification. We present the experimental setup in section 5 and results in section 6. Finally, we present our conclusions in section 7.

Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1018–1027, Osaka, Japan, December 11-17 2016.

2 Cognate detection

In this paper, we work with Swadesh lists (Swadesh, 1952) that are composed of meanings which are supposed to be resistant to lexical replacement and borrowing.

Meaning	Swedish	English	German
foot	fut (B)	fut (B)	fus (B)
belly	mag3 (N)	bEli (B)	baux (B)
to sew	si (F)	s3u (F)	nE3n (B)

Table 1: A excerpt of Swadesh list from Indo-European Lexical database for Swedish, German, and English for three meanings “foot”, “bell”, and “to sew”. The lexical items are transcribed in ASJP alphabet which is given in table 2. The cognate class labels, indicated in parentheses, do not carry additional information across meanings.

Table 1 shows the cognate class of each lexical item. Within a meaning, if two lexical items belong to a same cognate class, then they are cognates otherwise, they are treated as non-cognates. For example, all word pairs in meaning “foot” belong to the same cognate class “B” and are cognates whereas, the word pairs for English and German are cognate in meaning class “belly” and are not cognate in the meaning class “to sew”. The task at hand is to correctly identify if two words from different languages belonging to a meaning class is cognate or not.

3 Convolutional Networks

In this section, we briefly describe some past work that uses CNNs for NLP tasks such as text classification and part-of-speech tagging. Then, we motivate the use of CNNs for cognate identification task.

The supervised approaches to cognate identification supply string similarity or phonetic similarity scores as features which might not capture all the information in two words. Character alignments extracted from minimum-edit-distance are used to train a linear classifier; and, the alignment features are further augmented by the context to capture processes of sound correspondences between two words (Bergsma and Kondrak, 2007; Ciobanu and Dinu, 2014). In a recent paper, Ciobanu and Dinu (2014) use character alignments from word pairs (extracted from a etymological dictionary) as features to train and test SVM classifiers. This method seems to require thousands of word pairs; and, might not be practically feasible in a low-data scenario. The approach of Bergsma and Kondrak (2007) which learns the alignment weights of characters requires monolingual corpora for source and target languages which is not available for many of the world’s languages.

In this context, CNNs can be an alternative way to avoid explicit feature engineering through similarity computation and can extract relevant features from a raw word pair. Also, CNNs do not require explicit character alignment since the weights for non-monotonic shared features between two words can be learned through back-propagation.

3.1 CNNs in NLP

Collobert et al. (2011) proposed ConvNets for NLP tasks in 2011 and have been applied for sentence classification (Kim, 2014; Johnson and Zhang, 2015; Kalchbrenner et al., 2014; Zhang et al., 2015), part-of-speech tagging (dos Santos and Gatti, 2014), and information retrieval (Shen et al., 2014).

Santos and Zadrozny (2014) use character embeddings in conjunction with word embeddings to train a convolutional architecture for the classification of short texts. The authors find that their architecture performs better than the systems reported in Socher et al. (2013). In a recent work, Zhang et al. (2015) treat documents as a sequence of characters and transform each document into a sequence of one-hot character vectors. The authors designed and trained two nine layer convolutional networks for the purpose of text classification. The authors report competitive or state-of-the art performance on a wide range of text classification datasets.

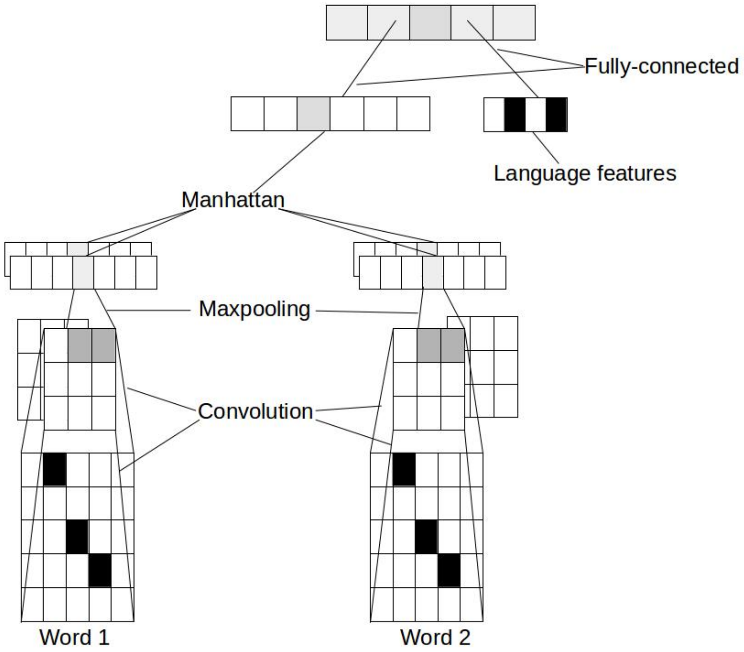

Figure 1: Illustration of Manhattan Siamese Convolutional network. We show the language features as a separate vector. Hot cells are shown in black whereas, real-valued cells are shown in grayscale.

3.2 Siamese Manhattan CNNs

Formally, we define the supervised problem setting where each training example x_i consists of two words x_{ia}, x_{ib} and a label $y_i \in \{0, 1\}$. Each phoneme $x_{iap} \in \mathbb{R}^k$ is a k-dimensional vector. A word is zero-padded or clipped at a pre-determined length n when necessary. A word x_{ia} of length n is represented as:

$$x_{ia} = x_{ia1} \oplus x_{ia2} \oplus \ldots \oplus x_{ian} \tag{1}$$

where, $\oplus$ is a concatenation operator. A convolution operation has a filter $W \in \mathbb{R}^{hm}$ where $h \leq k$ and $m < n$. The window size m defines the size of the filter. The feature map $C \in \mathbb{R}^{pq}$ where, $p = k - h + 1, q = n - m + 1$ is formed by convoluting the filter W with word x_{ia}. A max-pooling operation takes as input $C \in \mathbb{R}^{pq}$ feature map and applies the $max(C_{s \times t})$ to generate a feature $\hat{C} \in \mathbb{R}^{\lfloor p/s \rfloor \lfloor q/t \rfloor}$. The features generated by multiple filters are passed to a sigmoid function $\frac{1}{(1+exp(-x))}$ that computes the probabilities for y_i.

In the original siamese architecture proposed by Chopra et al. (2005), the weights are tied for each input x_{ia}, x_{ib}. The ℓ_2-norm (D) between the representations R_{ia}, R_{ib} computed using the shared convolutional networks of x_{ia}, x_{ib} and the label y_i is used to train a contrastive loss function $y_i \cdot D + (1 - y_i) \cdot max\{0, m - D\}$ where, m is a constant that can be tuned during training.

In this paper, we extend the siamese architecture to include an element-wise absolute difference layer which can then be stacked with multiple fully-connected layers. The final layer would be a sigmoid layer for binary classes. The idea behind this step is to push the CNNs to learn the phonological differences during training. The absolute difference $(-)$ operation resembles ℓ_1 norm and is defined as

$$M_{iab} = |R_{ia} - R_{ib}| \tag{2}$$

where, $M_{iab} \in \mathbb{R}^r$ and r is the length of the representation vector at the end of convolutional layer. Hence, we call this architecture as Manhattan CNN. Parts of this architecture is shown in figure 1.

3.3 Phoneme encodings

Santos and Zadrozny (2014) train character embeddings for boosting their short text classification system based on CNNs. However, the cognate identification task typically deals with short word lists (~ 200) and short words (~ 5). However, many of the languages such as those studied in this paper do not have enough corpora to train character embeddings. Due to these reasons, we use 1-hot and hand-crafted phoneme encodings to train our convolutional networks.

1-hot phoneme CNN In this representation, each phoneme p is represented as 1-hot vector $\in \mathbb{R}^{|P|}$ where, P is the set of phonemes in a language family. Each word is either zero-padded to attain a length of n or clipped if the length exceeds a fixed length. We use the phonetic alphabet developed by Brown et al. (2008)[1] – for computerized historical linguistics – in our experiments. The ASJP alphabet and its phonetic properties are given in table 2. Word delimiters are represented by **0** vectors. We refer this architecture as CharCNN.

Phonetic features CNN In this representation, we encode each phoneme p as a $1/0$ vector of phonetic features. The description of phonetic properties of each phoneme is given in table 2. The features are ordered as they appear in the description of the alphabet in Brown et al. (2008). The first motivation behind this approach is to test if we can use the phonetic information (that is available with the word lists) for cognate identification. The second motivation is to test if CNNs can directly learn the patterns of sound change from underlying phonetic representations for the purpose of cognate identification. We refer this architecture as PhoneticCNN.

Features	p	b	f	v	m	8	4	t	d	s	z	c	n	S	Z	C	j	T	5	k	g	x	N	q	G	X	7	h	l	L	w	y	r	!	V
Voiced	0	1	0	1	1	1	1	0	1	0	1	1	1	0	1	0	1	1	0	0	1	1	1	0	1	1	0	1	1	1	1	1	1	1	1
Labial	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Dental	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Alveolar	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Palatal/Post-alveolar	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Velar	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0
Uvular	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0
Glottal	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
Stop	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	0	1	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0
Fricative	1	1	1	1	0	1	0	0	0	1	1	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0	0	0	0	0	0
Affricate	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Nasal	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Click	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Approximant	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0
Lateral	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
Rhotic	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Table 2: The ASJP alphabet is given in columns $2 - 35$ and the phonetic value of each symbol in the ASJP alphabet. Each phoneme is a multi-hot vector of fixed dimension 16.

3.4 Language features

One major limitation of previous work in cognate identification is that the weight training of word similarity features is not performed jointly with language relatedness information. We present an architecture to learn the phonological similarity jointly with weighted language relatedness. We extend the Manhattan architecture to include language relatedness information during training.

Some languages share more cognate pairs than other language pairs due to genetic relatedness. We can train the model to learn language relatedness jointly with phonological relatedness by representing the languages as 2-hot vector. Formally, two words x_{ia}, x_{ib} belong to different languages $l_a, l_b \in$ language set L is represented as 2-hot vector $\in \mathbb{R}^{|L|}$ which is concatenated with the learned representation M_{iab}. The concatenated vector is then passed to a fully-connected layer whose output is then passed to a sigmoid layer. All our models are trained with binary cross-entropy loss function defined as $-(y_i \cdot log(s_i) + (1 - y_i) \cdot log(s_i))$ where s_i is the score for an instance i at the final sigmoid layer. The architecture with language features and the fully connected layer is shown in figure 1.

[1] Known as Automated Similarity Judgment Program; `asjp.clld.org`. The website provides 40 length word lists for more than 4000 of the world's languages and lists of length 100 for some languages. Very few word lists have cognate judgments such as Mayan language family which we include in this work.

We observe that including language relatedness (phylogeny) information seems to be quite challenging. For instance, the work of Bouchard-Côté et al. (2013) uses the inferred phylogeny of Austronesian languages (Greenhill and Gray, 2009) and do not infer the phylogeny themselves. In the case of Indo-European, Bouckaert et al. (2012) infer a Indo-European phylogeny from the cognacy information encoded for 200-word Swadesh lists and do not infer the cognacy judgments jointly with phylogeny.[2]

Using the Indo-European phylogeny information given in Glottolog (Nordhoff and Hammarström, 2011) can be circular since the cognacy judgments used by Bouckaert et al. (2012) are also used by human experts to derive the phylogeny information given in Glottolog. Therefore, we include the language information that is available with the word lists and hypothesize that a fully connected neural network layer can learn the weights of the language features jointly with the phonological representations generated by siamese CNNs through back-propagation.

4 Related work

The past work on cognate identification is mostly based on supervised approaches such as (Hauer and Kondrak, 2011; Bergsma and Kondrak, 2007; Inkpen et al., 2005) and graphical model approaches (Bouchard-Côté et al., 2013). In a different line of work, Kondrak (2000) and List (2012) employ linguistically motivated phoneme correspondence weights for computing the similarity between word pairs.

Inkpen et al. (2005) test the efficacy of different machine learning algorithms to determine if a pair of words are cognates or not. They use various orthographic similarity measures as features for the machine learning algorithms. They train and test their models on word pairs extracted from parallel texts and English-French cognate list; and find that there is no single machine learning algorithm that is good at both the datasets.

Hauer and Kondrak (2011) motivate a SVM classifier for the purpose of clustering word pairs within a meaning. They supply string similarity measures as features for their SVM classifier and then use the trained model to score the extracted word pairs from the testing part of their data. In this paper, we compare our neural network models against their classifier.

Ciobanu and Dinu (2014) test if character alignments extracted from Longest Common Subsequence alignments can be employed for the purpose of pair-wise cognate detection. They train a binary SVM classifier using the multi-gram character alignments as features for four pairs of Romance languages: Romanian-French, Romanian-Italian, Romanian-Spanish, and Romanian-Portuguese. They find that the SVM classifier trained on character alignments performs better than the orthographic similarity measures such as Edit distance, Longest Common Subsequence Ratio, and number of common bigrams.

Bouchard-Côté et al. (2013) employ a graphical model to reconstruct the word forms in Proto-Austronesian using Swadesh lists. They find that the inferred proto-forms largely agree with the reconstructed proto-forms. However, their method requires cognate information and the phylogeny of the language family to be known beforehand. In this article, we also experiment with a subset of Austronesian language family.

5 Experiments

5.1 Hyperparameters and training

The number of feature maps in a convolutional layer is fixed at 10. The architecture features a max-pooling layer that halves the output of the previous convolutional layer. We used the dropout technique with 0.5 probability (Srivastava et al., 2014) to prevent a fully-connected layer from over-fitting. A fully connected layer is trained with ReLU non-linearity ($max(0, x)$). The filter width m is fixed at 2 for 1-hot phoneme CNNs and 3 for phonetic feature CNNs. The filter length h is fixed as the size of $|P|$ for 1-hot phoneme CNNs and 2 for phonetic feature CNNs. The word length parameter n is fixed at 10. We used adadelta optimizer (Zeiler, 2012) with learning rate of 1.0, $\rho = 0.95$, and $\epsilon = 10^{-6}$. We fixed the mini-batch size to 128 in all our experiments. Both our architectures are relatively shallow – only

[2] The Indo-European work also includes higher level subgrouping information as priors to infer the divergence ages along the root and internal nodes of the phylogeny.

three layers – as compared to the text classification architecture of Zhang et al. (2015). We trained all our networks using Keras (Chollet, 2015) and Tensorflow (Abadi et al., 2016).

5.2 Datasets

We evaluate the performance of phoneme CNNs on three different language families: Austronesian, Indo-European, and Mayan.

Austronesian The Austronesian Basic Vocabulary Database[3] has word lists for 210 concepts in 378 languages. The database also has a cognacy judgment for each word. However, the database is not in an uniform transcription. Hence, we semi-automatically processed the words and converted a subset of 100 languages into uniform ASJP alphabet. We extracted a total of $525,941$ word pairs from the processed data of which $167,676$ are cognates.

Indo-European The second dataset comes from the Indo-European Lexical database which was originally created by Dyen et al. (1992) and curated by Michael Dunn.[4] The database is transcribed in a mix of International Phonetic Alphabet (IPA) and Romanized IPA. The database has word lists for 207 concepts in 139 languages. We extracted word lists for only those languages which are in phonemic transcription in more than 80% of the concepts. This filtering step leaves us with a total of $326,758$ word pairs for 52 languages of which $83,403$ are cognates.

Mayan The third dataset comes from the Mayan language family (Wichmann and Holman, 2013) that is spoken in Meso-America. This dataset has word lists in ASJP format for 100 concepts in 30 languages. We extracted $63,028$ word pairs from the dataset out of which $22,756$ are cognates.

Family	Training		Testing		$\|P\|$	$\|L\|$	Avg. # Cognate Classes
	Non-Cognates	Cognates	Non-Cognates	Cognates			
Austronesian	$244,978$	$125,018$	$113,287$	$42,658$	35	100	22.095
Indo-European	$162,818$	$62,120$	$80,537$	$21,283$	38	52	12.21
Mayan	$17,740$	$10,482$	$8,047$	$4,297$	33	30	8.58

Table 3: The number of positive and negative examples in training and testing datasets is given for each family. The size of the alphabet ($|P|$), number of languages ($|L|$) and, the average number of cognate classes per concept for each family.

5.3 Evaluation metrics

The performance of the baseline and the different CNN models is evaluated using Accuracy (ACC) and F-score. Given W word pairs, Accuracy is defined as the number of word pairs that have been assigned the correct labels (both cognate and non-cognate) divided by W. The F-score is defined as the harmonic mean of the Precision (P) and Recall (R) ($\frac{2PR}{P+R}$).

5.4 Baseline

We compare the performance of CNNs against the SVM classifier system trained on the following features from Hauer and Kondrak (2011). We used a linear kernel and optimized the SVM hyperparameter (C) through ten-fold cross-validation and grid search on the training data.

- Edit distance.
- Common number of bigrams.
- Length of longest common prefix.
- Lengths of both the words.
- Absolute difference between lengths of words.

[3] http://language.psy.auckland.ac.nz/austronesian/ (Greenhill et al., 2008). We accessed the database on 09-12-2015.

[4] http://ielex.mpi.nl/

6 Results

For each family, we train our models on word pairs extracted from $\sim 70\%$ of the meanings and test on the remaining meanings. The details of the training and testing datasets are given in table 3. The results of our experiments are given in table 4.

Systems	Indo-European		Austronesian		Mayan	
	F-score	Accuracy	F-score	Accuracy	F-score	Accuracy
Baseline	80.1	78.92	77.1	76.54	81.3	80.96
PhoneticCNN	85.8	**86.6**	77.6	79.24	85.4	85.56
PhoneticCNN + Langs.	**86.1**	86.42	78.3	79.8	86.2	86.23
CharCNN	84.6	85.05	79.1	80.11	86.3	86.4
CharCNN + Langs.	85.7	86.03	**80.3**	**80.94**	**87.5**	**87.5**

Table 4: Accuracies and F-scores of different CNN models against the system of Hauer and Kondrak (2011). CNNs with language features are denoted with a suffix "+ Langs.".

All the CNN models perform better than the baseline across all the language families. The PhoneticCNNs perform better than the CharCNN only on the Indo-European language family. In the case of Austronesian language family, joint training of language features improve the performance over baseline. This is reasonable since the Austronesian language family is spread over a wide range of geographical area spreading from Madagascar to Hawaii. The joint training of language features also improves the accuracy and F-score for Mayan language family.

CharCNN performs the best on the Mayan language family. One reason for this could be that the Mayan language family is a geographically proximal family and does not exhibit great amount of phonological divergence. Moreover, the Mayan language family shows less number of average cognate classes per concept as compared to Austronesian or Indo-European (cf. table 3) which can interpreted as a measure of genetic closeness within a family. In the case of Indo-European, the phonetic CNNs trained jointly with language information perform the best.

6.1 Do CNNs work with small training sets?

Zhang et al. (2015) note that CNNs require large amount of data for training. We test this hypothesis by training our CNNs on a smaller subset of 20 concepts. The results of our experiments are given in table 5.

Systems	Indo-European		Austronesian		Mayan	
	F-score	Accuracy	F-score	Accuracy	F-score	Accuracy
Baseline	81.8	81.05	**77.9**	**77.7**	80.5	80.02
PhoneticCNN	83	**84**	73.6	75.86	84.6	84.64
PhoneticCNN + Langs.	**83**	83.78	73.1	75.82	84.1	84.25
CharCNN	79.6	81.62	74.3	76.69	**85.6**	**85.55**
CharCNN + Langs.	80.9	82.61	76.0	77.84	81.2	81.36

Table 5: Accuracies and F-scores of different CNN models trained on 20 meanings in the training data.

In the case of Indo-European and Mayan, the CNNs perform better than the baseline whereas for Austronesian the CNNs do not outperform the baseline system. The results for Indo-European and Mayan (cf. table 5) are similar to that of the results reported in table 4. That is, the CharCNN system performs the best for Mayan language family, while the PhoneticCNN system performs the best for the Indo-European language family. Surprisingly, for the Austronesian family, the baseline system performs better at F-score than the top-performing system for this language family in table 4, namely the CharCNN (with language features); the Accuracy measure of the Baseline system is also higher, but the difference is not statistically significant. The reason for this could be that there is not enough information in the 20 meanings to learn phonological similarity for 100 languages.

The results for Mayan family suggests that the CharCNN can be used with small datasets for a closely related language family. We believe that this is an important result due to the abundance of small number of language families in the world.

To support our claim, we cite family size numbers from Glottolog[5] which show that there are about 50 language families of size between 10 and 100. Due to this reason, we claim that a cognate identification system that can perform well on geographically proximal, closely related languages is useful for identifying cognates, which, in turn, can be used for inferring phylogenies of under-studied language families.

7 Conclusion

In this article, we proposed siamese CNNs for cognate identification and compared it against a SVM classifier trained on orthographic similarities. Our results suggest that CharCNNs and PhoneticCNNs can be used for the purpose of cognate identification. Our results on Mayan language families suggest that CNNs can be applied for NLP tasks in closely related languages or varieties. The language features improve the performance of CNNs across all the language families.

The performance of CharCNNs suggest that deep learning can be applied for small datasets (language families). Many deep learning systems reported in the NLP literature require huge amount of training data. Here, we show that handcrafted embedding and 1-hot encodings can learn useful representations from raw words for capturing phonological similarities between a word pair.

In the future, we hope to apply CNNs for more language families of the world for the purpose of cognate identification and phylogenetic inference.

Acknowledgements

I thank the reviewers for the useful comments which helped improve the paper. This work has been supported by the ERC Advanded Grant 324246 EVOLAEMP, which is gratefully acknowledged. I thank Simon Greenhill for the permission to use the Austronesian data in the experiments. The data for the experiments was processed by Johann-Mattis List and Pavel Sofroniev. I thank Çağrı Çöltekin for the comments on the initial draft that helped improved the paper.

References

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*.

Shane Bergsma and Grzegorz Kondrak. 2007. Alignment-based discriminative string similarity. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 656.

Alexandre Bouchard-Côté, David Hall, Thomas L. Griffiths, and Dan Klein. 2013. Automated reconstruction of ancient languages using probabilistic models of sound change. *Proceedings of the National Academy of Sciences*, 110(11):4224–4229.

Remco Bouckaert, Philippe Lemey, Michael Dunn, Simon J. Greenhill, Alexander V. Alekseyenko, Alexei J. Drummond, Russell D. Gray, Marc A. Suchard, and Quentin D. Atkinson. 2012. Mapping the origins and expansion of the Indo-European language family. *Science*, 337(6097):957–960.

Cecil H. Brown, Eric W. Holman, Søren Wichmann, and Viveka Velupillai. 2008. Automated classification of the world's languages: A description of the method and preliminary results. *Sprachtypologie und Universalienforschung*, 61(4):285–308.

Will Chang, Chundra Cathcart, David Hall, and Andrew Garrett. 2015. Ancestry-constrained phylogenetic analysis supports the Indo-European steppe hypothesis. *Language*, 91(1):194–244.

François Chollet. 2015. Keras. *GitHub repository: https://github. com/fchollet/keras*.

[5] `http://glottolog.org/glottolog/family`. Accessed on 15-07-2016.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 539–546. IEEE.

Alina Maria Ciobanu and Liviu P Dinu. 2014. Automatic detection of cognates using orthographic alignment. In *ACL (2)*, pages 99–105.

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

Cícero Nogueira dos Santos and Maira Gatti. 2014. Deep convolutional neural networks for sentiment analysis of short texts. In *COLING*, pages 69–78.

Isidore Dyen, Joseph B. Kruskal, and Paul Black. 1992. An Indo-European classification: A lexicostatistical experiment. *Transactions of the American Philosophical Society*, 82(5):1–132.

Simon J. Greenhill and Russell D. Gray. 2009. Austronesian language phylogenies: Myths and misconceptions about Bayesian computational methods. *Austronesian Historical Linguistics and Culture History: A Festschrift for Robert Blust*, pages 375–397.

Simon J. Greenhill, Robert Blust, and Russell D. Gray. 2008. The Austronesian basic vocabulary database: from bioinformatics to lexomics. *Evolutionary Bioinformatics Online*, 4:271–283.

Bradley Hauer and Grzegorz Kondrak. 2011. Clustering semantically equivalent words into cognate sets in multilingual lists. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 865–873, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

Diana Inkpen, Oana Frunza, and Grzegorz Kondrak. 2005. Automatic identification of cognates and false friends in French and English. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 251–257.

Rie Johnson and Tong Zhang. 2015. Effective use of word order for text categorization with convolutional neural networks. In *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015*, pages 103–112.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, June.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October. Association for Computational Linguistics.

Grzegorz Kondrak. 2000. A new algorithm for the alignment of phonetic sequences. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 288–295.

Grzegorz Kondrak. 2009. Identification of cognates and recurrent sound correspondences in word lists. *Traitement Automatique des Langues et Langues Anciennes*, 50(2):201–235, October.

Johann-Mattis List. 2012. LexStat: Automatic detection of cognates in multilingual wordlists. In *Proceedings of the EACL 2012 Joint Workshop of LINGVIS & UNCLH*, pages 117–125, Avignon, France, April. Association for Computational Linguistics.

Sebastian Nordhoff and Harald Hammarström. 2011. Glottolog/Langdoc: Defining dialects, languages, and language families as collections of resources. In *Proceedings of the First International Workshop on Linked Science*, volume 783.

Taraka Rama. 2016. Ancestry sampling for indo-european phylogeny and dates.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Morris Swadesh. 1952. Lexico-statistic dating of prehistoric ethnic contacts: with special reference to North American Indians and Eskimos. *Proceedings of the American philosophical society*, 96(4):452–463.

Søren Wichmann and Eric W Holman. 2013. Languages with longer words have more lexical change. In *Approaches to Measuring Linguistic Differences*, pages 249–281. Mouton de Gruyter.

Matthew D Zeiler. 2012. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc.