Learning character-wise text representations with Elman nets

Grzegorz Chrupała G.CHRUPALA@UVT.NL
Tilburg University, Warandelaan 2, 5037 AB Tilburg, Netherlands

Keywords: recurrent neural networks, embeddings, text segmentation, text normalization

Simple recurrent networks (SRNs) were introduced by Elman (1990) in order to model temporal structures in general and sequential structure in language in particular. More recently, SRN-based language models have become practical to train on large datasets and shown to outperform n-gram language models for speech recognition (Mikolov et al., 2010). In a parallel development, word embeddings induced using feed-forward neural networks have proved to provide expressive and informative features for many language processing tasks (Collobert et al., 2011; Socher et al., 2012).

The majority of representations of text used in computational linguistics are based on words as the smallest units. Words are not always the most appropriate atomic unit: this is the case for languages where orthographic words correspond to whole English phrases or sentences. It is equally the case when the text analysis task needs to be performed at character level: for example when segmenting text into tokens or when normalizing corrupted text into its canonical form.

In this work we propose a mechanism to learn *character-level* representations of text. Our representations are low-dimensional real-valued embeddings which form an abstraction over the character string prior to each position in a stream of characters. They correspond to the activation of the hidden layer in a simple recurrent neural network. The network is trained as a language model: it is sequentially presented with each character in a string (encoded using a one-hot vector) and learns to predict the next character in the sequence. The representation of history is stored in a limited number of hidden units (we use 400), which forces the network to create a compressed and abstract representation rather than memorize verbatim strings. After training the network on large amounts on unlabeled text, it can be run on unseen character sequences, and activations of its hidden layer units can be recorded at each position and used as features in a supervised learning model.

We use these representation as input features (in addition to character n-grams) for text analysis tasks: learning to detect and label programming language code samples embedded in natural language text (Chrupała, 2013), learning to segment text into words and sentences (Evang et al., 2013) and learning to translate non-canonical user generated contents into a normalized form (Chrupała, 2014). For all tasks and languages we obtain consistent performance boosts in comparison with using only character n-gram features, with relative error reductions ranging from around 12% for English tweet normalization to around 85% for Dutch word and sentence segmentation.

References

Chrupała, G. (2013). Text segmentation with character-level text embeddings. *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Chrupała, G. (2014). Normalizing tweets with edit scripts and recurrent neural embeddings. *ACL*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, *12*, 2493–2537.

Elman, J. L. (1990). Finding structure in time. *Cognitive science*, *14*, 179–211.

Evang, K., Basile, V., Chrupała, G., & Bos, J. (2013). Elephant: Sequence labeling for word and sentence segmentation. *EMNLP*.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., & Khudanpur, S. (2010). Recurrent neural network based language model. *INTERSPEECH*.

Socher, R., Huval, B., Manning, C. D., & Ng, A. Y. (2012). Semantic compositionality through recursive matrix-vector spaces. *EMNLP-CoNLL*.