LDA-T3114 Introduction to Deep Learning (2019): Final Project Report

firstname lastname
organisation
e-mail

Abstract

This project report describes the system submitted as the final assignment for the course *Introduction to Deep Learning*. The system is intended for the morphological reinflection task organised by SIGMORPHON, and attempts to follow the encoder-decoder system developed by Kann and Schütze (2016). While remarkably more simple and modest in implementation compared to the original design, the final model eventually achieved satisfactory results and proved that the base architecture works very well.

1 Introduction

The topic of this final project was morphological reinflection on three languages. I tried to follow the general idea of the MED system (*Morphological Encoder-Decoder*), which Kann and Schütze (2016) presented for the SIGMORPHON 2016 Shared Task on Morphological Reinflection. Their system is an encoder-decoder recurrent neural network (RNN), and it proved to be the all-around best performing system in the shared task. All in all, recurrent neural network architectures consistently performed best at the task, establishing a strong state of the art in morphological reinflection (Cotterell et al., 2016).

Morphological reinflection differs from the more traditional basic inflection task in that instead of generating an inflectional paradigm from a given lemma, the desired word form should be generated from a different, already inflected form. An example of morphological reinflection in the Finnish language would be generating the singular genitive case form *hatun* from the plural transitive *hatuiksi*. This task involves analysis of the source word, together with synthesis of the target word (Cotterell et al., 2016), which can be challenging with a limited data.

While the base structure attempts to mimic Kann and Schütze's work, my system is much smaller in scale: not only in terms of languages and data that were used, but also from the technical point of view. Nevertheless, the results achieved by the final model were mostly adequate enough, and it gave some concrete evidence of how powerful a morphological reinflection system based on neural networks can be.

2 Model

The system uses neural encoder-decoder models to predict correct word forms. The encoder, which is a bidirectional gated recurrent neural network, is given an input in the form of a single sequence that consists of the letters of the source form and the morphological tags of the target form. In task 2, source tags are also included in the input sequence. This sequence of vectors is encoded into a fixed-length context vector that is then fed to the decoder. Similar to the encoder, the decoder is also a gated RNN (GRU). The decoder uses the context vector and the target vector to predict the final output.

In the original system by Kann and Schütze (2016), a soft attention mechanism was used when calculating the context vector. Due to time constraints, this feature is not implemented in my model. To compensate for the lack of attention, the number of hidden units is increased to 200, while in the original system this number was 100. Similarly, the embedding dimension is increased to 200. In training, a minibatch size of 20 is used. One model is trained for each language, and the number of training epochs is 20. The hyperparameters are the same for every model: they were not tuned to any specific task or language during the experimentation. The system is implemented in Pytorch (Paszke et al., 2017).

3 Data

All the datasets are from the SIGMORPHON 2016 Shared Task [1]. Three languages were used: Finnish, German, and Navajo. Every language had its own training and development sets, and two test sets: one with the gold standard word form included, and one without it. Furthermore, every task had separate datasets. Thus, the amount of different individual data files was 36 in total.

Special start and end symbols were added to all possible inputs. For example, a valid input for Finnish would be *<w> a a l t o pos=N case=TRANS num=SG </w>*, and the correct corresponding system output should be *<w> a a l l o k s i </w>*. All possible input tags, output tags, and individual characters for the three languages were combined into one index library. This index library also included the start and end symbols, and a special padding character *<pad>*.

The structure of the datasets depends on the task. For task 1, the system input contains the source lemma and the target tags, and the output to be generated is the target inflected form. Task 2 gives an inflected source word form instead of a lemma, and source tags in addition to target tags. Task 3 again takes one step further and gives the inflected source form accompanied only by the target tags. This makes the tasks increasingly challenging, as the system needs to learn from more limited data.

In addition to the different tasks, the SIGMORPHON 2016 Shared Task was also split into three separate tracks. These tracks differ in the information that is available for the system: the restricted track limits the available training and development data to the current task, while the standard track allows the system to also utilise the corresponding datasets of all lower-numbered tasks. For example, the restricted track only allows task 2 data to be used when solving task 2, but the standard track of the same task also allows the system to include task 1 data. There was also a third track, which is a bonus track: all available data per language can be used, including monolingual data in the form of Wikipedia dumps, provided by the task organisers (Cotterell et al., 2016). For simplicity's sake, I only use the datasets pertinent to the task at hand: this essentially means that my system only makes use of the restricted track of the shared task.

[1] https://github.com/ryancotterell/sigmorphon2016/tree/master/data/

Language	Task 1	Task 2	Task 3
Finnish	40.10%	34.78%	35.12%
German	46.35%	43.01%	43.04%
Navajo	73.36%	55.12%	58.00%

Table 1: Exact-match accuracy per language and task.

Language	Task 1	Task 2	Task 3
Finnish	64.45%	59.59%	56.95%
German	89.44%	87.62%	80.13%
Navajo	53.06%	47.59%	44.96%

Table 2: SIGMORPHON 2016 Shared Task accuracy results for the baseline system (Cotterell et al., 2016).

4 Experiments

The experiments were executed for all languages in one run, and every task was handled separately. In evaluation, when the system generates the predicted output word forms, a maximum length of 30 is used as the limit for a possible word length. As was mentioned earlier, the number of training iterations was fixed to 20 epochs—I experimented with different epoch sizes, with the largest being 50, but the learning process seemed to peak around the 20th iteration.

Teacher forcing was used to speed up the training process, meaning that the correct target word form was also included in the decoder's inputs. The decoder bases its predictions on the previous characters in the word: for example, if the input lemma is *<w> k o i r a* (cutting off the final end symbol), the system should correctly predict the following characters *k o i r a </w>*. That is, the concatenation of the context vector and the embedding for the input symbol *<w>* should yield the character *k*, then the concatenation of the context vector and the embedding for character *k* should predict the character *o*, and so on.

5 Results

The system was evaluated using the gold standard test sets provided by SIGMORPHON. The batch size was 20, but every word in the batch was evaluated individually. The evaluation was done using exact-match accuracy: the output predicted by the system was judged to be correct if and only if it matched the target string perfectly. Results after 20 epochs can be seen in Table 1. The final results were achieved with a learning rate of 0.001.

Out of the three languages used, the model for Navajo consistently achieved the highest accuracy. Finnish and German performed more poorly. This seems rather inconsistent with the results achieved by the SIGMORPHON 2016 Shared Task baseline system, which can be seen in Table 2. It should be noted, however, that the baseline results are achieved on the standard track test set, whereas my system only runs on the restricted track. Furthermore, the results reported by Kann and Schütze (2016) also suggest that the models for Finnish and German should outperform Navajo. It is perfectly possible—and probably even highly likely, given my set of skills—that my system does not work as intended, and the more subtle mistakes and errors in the code ultimately show up in the results. There are, however, clear differences in the morphology of the given languages: both Finnish and German are highly suffixing, while Navajo is a primarily prefixing language. It does therefore make sense that the models for Finnish and German show more similar results.

6 Conclusions and Discussion

The system performance is decent, but not nearly as good as one would generally expect from an RNN. There can be various reasons for this, one of which is my own limited knowledge and technical know-how about neural networks. The original MED system used an attention mechanism, which is lacking from my abridged version: this disparity, while not fully explaining the low accuracies, might have a slight impact on the performance.

Implementation of the system proved to be rather challenging, and even downright frustrating at times: the architecture of encoder-decoder neural networks was not too familiar to me beforehand, and it took me quite some time to understand all the different layers and necessary functions. As the project turned out to be more time-consuming than I initially expected, many features of the original system were eventually omitted. I was especially displeased with the fact that I did not have time to implement the attention mechanism—this might be something that I would like to add later, should I find the time to do so. However, all things considered, I feel like I learned a lot during this project, and seeing the system finally work and actually learn from the data was highly rewarding in the end.

References

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The SIGMORPHON 2016 Shared Task—Morphological Reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 10–22. Association for Computational Linguistics.

Katharina Kann and Hinrich Schütze. 2016. MED: The LMU System for the SIGMORPHON 2016 Shared Task on Morphological Reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 62–70. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS-W*.