AX Semantics' Submission to the SIGMORPHON 2019 Shared Task

Andreas Madsack and **Robert Weißgraeber**
AX Semantics, Stuttgart, Germany
{firstname.lastname}@ax-semantics.com

Abstract

This paper describes the AX Semantics' submission to the SIGMORPHON 2019 shared task on morphological reinflection. We implemented two systems, both tackling the task for all languages in one codebase, without any underlying language specific features. The first one is an encoder-decoder model using AllenNLP; the second system uses the same model modified by a custom trainer that trains only with the target language resources after a specific threshold. We especially focused on building an implementation using AllenNLP with out-of-the-box methods to facilitate easy operation and reuse.

1 Introduction

This paper describes our implementation and results for Task 1 of the 2019 Shared Task (McCarthy et al., 2019). The task is to generate inflected word forms given the lemma and a morphological feature specification (Kirov et al., 2018). See Figure 1 for an example in German, where a verb lemma is inflected according to the specified number, mood, tense and person.

`sehen` (V;IND;PST;3;PL) → `sahen`

Figure 1: Task 1 Example, German: putting the verb "sehen" into 3rd person past tense indicative plural.

In contrast to last year, where the training data was only in the respective target language, this year the given data consists of up to 10000 exemplars of one high resource language combined with up to 100 exemplars of a low resource language. The target language is the low resource language. The task is to use the high resource data to improve the inflection of the low resource language. Including the surprise language pairs the task consists of 99 language pairs.

2 Motivation

After participating last year (Madsack et al., 2018) we started to rebuild everything we needed for our production system using AllenNLP (Gardner et al., 2017). Our main goal here is reproducibility and full logging of everything as default. In our experience AllenNLP brings best practices that, while sometimes opinionated, are way better than building everything from scratch, and which we wanted to apply to this problem.

Our two systems represent our learning curve in the attempt to solve the given shared task. The first system is a solution entirely based on given AllenNLP components. The second system has a custom trainer that, only at the start, trains with all given training data for a pair and then continues only with the (low-resource) target language.

The source code of our submission can be found at: `https://301.ax/github-sigmorphon2019`

3 System 1 - softmax baseline in AllenNLP

Our first system is the soft-attention baseline rebuilt in AllenNLP. It basically serves as a starting point for our second system.

The model is an encoder-decoder (Cho et al., 2014) and is using the readily implemented version in AllenNLP (named SimpleSeq2Seq). We modified the model code to add accuracy and edit-distance metrics. The attention used is dot-product attention (Luong et al., 2015). All other hyper parameters are inspired by Wu et al. (2018) and shown in Table 1.

We trained two kinds of System 1. One with only low data as baseline and another with high and low data concatenated. All systems used here are character based and the input sequence is first the lemma followed by a next marker (we

Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 1–5
Florence, Italy. August 2, 2019

used a tabulator) followed by the morphological features as a string. One example input of the encoder looks like the following: `zmrzlina N;DAT;SG`. Besides, AllenNLP wraps inputs and target outputs with start and end markers.

parameter	value	
	System 1	System 2
embedding dimension	200	100
beam size	10	10
hidden size	400	200
number of hidden encoder layers	2	1
encoder dropout	0.4	0.3
optimizer	adam	adam

Table 1: hyper parameters for System 1 and System 2

4 System 2 - transfer learning

The second system uses the same encoder-decoder-model as System 1. The major modification is a trainer that first learns on all training data (high and low resource data) and after a threshold is reached continues learning only with the target language. This threshold marks the transfer learning point: The cross-lingual model is reused as a basis for training with the monolingual data.

The first 10 epochs are always trained with all training data. The switch to only training with low resource data happens after 5 epochs without training improvement. As metric for this improvement a lower loss on validation data is used. Most hyper parameters (Table 1) for System 2 were halved after some experimental evaluation. We did not do an exhaustive search of these parameters, so minor improvements with the help of hyperparameter optimization (e.g. using cross-validated grid search) are possible here.

Figure 3 shows a loss curve where training with only the target language Khakas (and without the high-resource language Bashkir) started at epoch 17. For comparison the loss curve of System 1 for the same language pair is shown in Figure 2. In this example System 2 gains a smaller loss than System 1 on the validation data - System 2 reaches with about 0.2 half as much loss as System 1 with about 0.4.

5 Results

The results for System 1 and System 2 shown in Table 2 and Table 3 together with the soft-attention

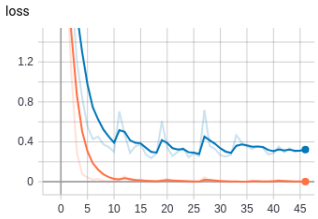

Figure 2: Loss for train (orange) and validation (blue) for **System 1** language pair "bashkir–khakas"

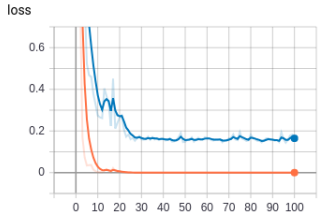

Figure 3: Loss for train (orange) and validation (blue) for **System 2** language pair "bashkir–khakas"

baseline from the organizers (Wu and Cotterell, 2019) are the unmodified results from the submission to the task. We found minor tooling mistakes on the surprise languages after the submission deadline which we didn't correct in the table.

In the trained models we can observe big differences in the accuracy for different language pairs. To better understand the results we trained a new version of System 1 with only the low data given and ignored the high-resource language data completely. As expected this version of System 1 performed worst in comparison to the other systems due to the lack of a sufficient amount of training data.

In general, the very low results on some pairs seem to be based on very different character sets and/or feature sets between the concerning language pairs. For example a language pair with a lot of different characters and different features is "bengali–greek". The amount of Greek data alone is not enough to train an encoder-decoder-model (see System 1 low results) and the data for Bengali doesn't help either way (see System 1, System 2 and baseline results).

Thus, the results indicate that a difference in features and/or character sets has a big impact on the usefulness of the high resource training data. For the character set a phonological mapping to a phonetic alphabet could improve on that issue.

language pair	characters in low	features not in high	System 1 (low)	System 1	System 2	Baseline (tune) (0-soft)
adyghe–kabardian	0	0	2	85	91	**93**
albanian–breton	2	7	0	10	11	**21**
arabic–classical-syriac	22	11	0	33	27	**52**
arabic–maltese	29	2	0	0	2	**16**
arabic–turkmen	30	3	6	8	12	**32**
armenian–kabardian	32	4	2	5	39	**68**
asturian–occitan	6	0	0	6	14	**47**
bashkir–azeri	32	10	0	19	11	**34**
bashkir–crimean-tatar	33	7	0	30	0	**51**
bashkir–kazakh	3	1	14	64	72	**76**
bashkir–khakas	3	3	2	*62*	***74***	**74**
bashkir–tatar	35	6	0	35	8	**37**
bashkir–turkmen	30	0	0	**52**	42	50
basque–kashubian	14	10	6	2	8	**20**
belarusian–old-irish	25	18	0	**4**	**4**	**4**
bengali–greek	82	16	0	0	0	**3.6**
bulgarian–old-church-slavonic	31	5	0	24	17	**40**
czech–kashubian	8	0	6	10	**52**	40
czech–latin	9	6	0	4.4	**6.7**	3.9
danish–middle-high-german	5	6	14	34	**70**	68
danish–middle-low-german	12	13	10	24	14	**36**
danish–north-frisian	3	10	0	7	20	**23**
danish–west-frisian	4	6	0	37	26	**48**
danish–yiddish	35	16	0	0	42	**44**
dutch–middle-high-german	2	5	10	50	**60**	54
dutch–middle-low-german	9	10	4	18	**38**	**38**
dutch–north-frisian	3	7	0	12	14	**21**
dutch–west-frisian	3	2	3	16	38	**43**
dutch–yiddish	35	13	0	-	-	**43**
english–murrinhpatha	0	7	0	12	**22**	12
english–north-frisian	4	12	0	2	19	**23**
english–west-frisian	5	8	0	19	33	**41**
estonian–ingrian	1	2	0	14	6	**30**
estonian–karelian	3	4	0	0	**46**	**46**
estonian–livonian	16	12	0	2	19	**25**
estonian–votic	3	1	3	14	17	**25**
finnish–ingrian	1	1	0	**36**	34	26
finnish–karelian	2	2	0	0	**52**	32
finnish–livonian	17	11	1	18	2	**25**
finnish–votic	4	2	2	27	**32**	22
french–occitan	3	1	0	24	**37**	33
german–middle-high-german	3	0	12	38	**72**	66
german–middle-low-german	10	7	8	2	20	**46**
german–yiddish	35	14	0	0	20	**46**
greek–bengali	*45*	*12*	1	*0*	*7*	***31***
hebrew–classical-syriac	22	10	0	48	32	**61**
hebrew–maltese	30	4	0	7	6	**16**
hindi–bengali	45	12	0	3	6	**35**
hungarian–ingrian	2	4	0	**24**	18	10
hungarian–karelian	3	8	2	0	**36**	30
hungarian–livonian	19	18	0	2	11	**19**
hungarian–votic	5	4	1	**17**	15	16
irish–breton	3	5	0	3	3	**19**
irish–cornish	3	8	0	2	**8**	**8**
irish–old-irish	1	12	0	2	**4**	0
irish–scottish-gaelic	5	2	0	42	26	**60**
italian–friulian	7	1	0	27	27	**33**
italian–ladin	2	3	1	13	23	**47**
italian–maltese	5	5	0	11	**16**	9
italian–neapolitan	2	2	6	**60**	48	41
kannada–telugu	23	1	20	44	**68**	60
kurmanji–sorani	9	12	0	2	0.8	**8.1**
latin–czech	17	8	0	0	9.1	**13.5**

Table 2: Left: Feature/character differences between language pairs. (in low, not in high language)
Right: Results (accuracy) for test data compared to baseline (Part 1)

language pair	characters in low	features not in high	System 1 (low)	System 1	System 2	Baseline (tune) (0-soft)
latvian–lithuanian	29	6	0	0.7	7.7	**10.9**
latvian–scottish-gaelic	11	0	0	30	**48**	**48**
persian–azeri	32	13	0	0	1	**23**
persian–pashto	15	9	0	0	1	**14**
polish–kashubian	6	0	6	48	**68**	66
polish–old-church-slavonic	57	1	0	10	0	**30**
portuguese–russian	36	15	0	0	0	**11.9**
romanian–latin	13	9	0	0	0.1	**4.5**
russian–old-church-slavonic	31	2	0	22	24	**32**
russian–portuguese	35	8	0	0.3	0.5	**32.3**
sanskrit–bengali	46	19	0	11	1	**21**
sanskrit–pashto	38	12	0	2	3	**7**
slovak–kashubian	9	1	2	22	40	**52**
slovene–old-saxon	7	6	0	4	6.7	**7.8**
sorani–irish	26	18	0	0.3	**3.3**	2.6
spanish–friulian	8	1	0	28	37	**38**
spanish–occitan	5	1	0	26	39	**50**
swahili–quechua	5	34	0	0	0.2	**3**
turkish–azeri	3	1	0	60	64	**66**
turkish–crimean-tatar	2	3	0	69	**74**	65
turkish–kazakh	31	1	18	54	68	**74**
turkish–khakas	25	3	2	68	54	**78**
turkish–tatar	4	2	0	**79**	68	69
turkish–turkmen	4	0	0	56	**86**	80
urdu–bengali	45	9	0	3	5	**30**
urdu–old-english	38	6	0.2	0.3	0.1	**8**
uzbek–azeri	12	6	0	5	4	**27**
uzbek–crimean-tatar	13	8	0	0	0	**13**
uzbek–kazakh	31	1	22	12	46	**56**
uzbek–khakas	25	3	0	10	28	**76**
uzbek–tatar	16	7	0	1	2	**21**
uzbek–turkmen	11	0	0	8	16	**36**
welsh–breton	6	8	0	17	20	**34**
welsh–cornish	4	11	2	0	12	**26**
welsh–old-irish	8	18	2	2	4	**8**
welsh–scottish-gaelic	12	11	0	20	16	**28**
zulu–swahili	0	19	0	0	19	**36**

Table 3: Left: Feature/character differences between language pairs. (in low, not in high language) Right: Results (accuracy) for test data compared to baseline (Part 2)

6 Conclusion

Our continual goal is to improve our morphology system component in our Natural Language Generation SaaS (Weißgraeber and Madsack, 2017).

In our production setup the System 1 described above competes against a handcrafted morphology and a reasonable lexicon (which were not used for the Shared Task). This handcrafted morphology together with the lexicon is always better on very regular part of speech (POS) types (i.e. German adjectives). Therefore not for every language POS combination a system shown here is used in our production NLG inflection system. For every language and POS type we evaluate which solution fits best.

AllenNLP successfully helped us to reproduce the same results even with newer versions of libraries (i.e. PyTorch, CUDA, Python), which is an important quality for our NLG system.

References

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. 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, Doha, Qatar. Association for Computational Linguistics.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Christo Kirov, Ryan Cotterell, John Sylak-Glassman, Géraldine Walther, Ekaterina Vylomova, Patrick

Xia, Manaal Faruqui, Sebastian J. Mielke, Arya McCarthy, Sandra Kübler, David Yarowsky, Jason Eisner, and Mans Hulden. 2018. UniMorph 2.0: Universal Morphology. In *Proceedings of the 11th Language Resources and Evaluation Conference*, Miyazaki, Japan. European Language Resource Association.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Andreas Madsack, Alessia Cavallo, Johanna Heininger, and Robert Weißgraeber. 2018. AX semantics' submission to the CoNLL–SIGMORPHON 2018 shared task. In *Proceedings of the CoNLL–SIGMORPHON 2018 Shared Task: Universal Morphological Reinflection*, pages 43–47, Brussels. Association for Computational Linguistics.

Arya D. McCarthy, Ekaterina Vylomova, Shijie Wu, Chaitanya Malaviya, Lawrence Wolf-Sonkin, Garrett Nicolai, Christo Kirov, Miikka Silfverberg, Sebastian Mielke, Jeffrey Heinz, Ryan Cotterell, and Mans Hulden. 2019. The SIGMORPHON 2019 shared task: Crosslinguality and context in morphology. In *Proceedings of the 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Florence, Italy. Association for Computational Linguistics.

Robert Weißgraeber and Andreas Madsack. 2017. A working, non-trivial, topically indifferent nlg system for 17 languages. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 156–157. Association for Computational Linguistics.

Shijie Wu and Ryan Cotterell. 2019. Exact hard monotonic attention for character-level transduction. *arXiv preprint arXiv:1905.06319*.

Shijie Wu, Pamela Shapiro, and Ryan Cotterell. 2018. Hard non-monotonic attention for character-level transduction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4425–4438, Brussels, Belgium. Association for Computational Linguistics.