Neural Morphological Generators for Hungarian

László János Laki, Noémi Ligeti-Nagy, Noémi Vadász, Zijian Győző Yang

Hungarian Research Centre for Linguistics
H-1068 Budapest, Benczúr u. 33.
{surname.forename}@nytud.hu

Abstract. Here we present a set of morphological generators for Hungarian that generate surface forms from emMorph and Universal Dependencies (UD) morphological tags with high accuracy. We experimented with two approaches: first, neural machine translation models were trained based on the morphological analysis as the source format and the corresponding surface form as the target format. Second, we tackled the problem as a text generation task, where the morphological analysis is followed by the correct word form. The corpus we used is a normalised version of Webcorpus 2.0 (Nemeskey, 2020). Marian MT proved to produce the best results, thus we evaluated its output manually on NerKor (Simon and Vadász, 2021). Our analysis shows that the generator achieves a high accuracy of 96.27% in the case of emMorph and 94.94% in the case of UD. After manual evaluation, we counted a more concise accuracy, which is 99.43% (emMorph) and 98.69% (UD). This model may be used for several NLP tasks, such as anonymisation and terminology translation.
Keywords: morphological generator, neural generator, emMorph, Universal Dependencies morphology

1 Introduction

A morphological generator is a program that performs the task of generating a word form based on the morphological analysis. Morphological generation may be considered an opposite task of morphological analysis. Here, given the description of a word in terms of number, category, stem, and so on, the original word is retrieved. For example, if root = *go*, part of speech = *verb*, tense= *present*, and if it occurs along with a third person and singular subject, then a morphological generator would generate its surface form, *goes*.

Despite being a core task of paramount importance for many NLP tasks, the number and the research of morphological generators is far below that of morphological analysers. Morphological generators, if built, are usually built together with morphological analysers. Morphological generation is a crucial task for languages with rich morphology in tasks like anonymisation and pseudonymisation, where the removed personal identifier must be replaced with another one, but in the correct surface form.

After a brief overview of the related works, we will present our experiments and then show the results of the evaluation and error analysis.

Our finetuned mT5 models are available on our Hugging Face page. The name of the emmorph tagset based model is *NYTK/morphological-generator-emmorph-mt5-hungarian*[1], while the name of the model for the UD tagset is *NYTK/morphological-generator-ud-mt5-hungarian*[2].

2 Related works

Minnen et al. (2000) presents a fast and robust morphological generator for English based on finite-state techniques. They illustrate the relevance of their generator on the automatic simplification of English newspaper texts.

As morphological generation is more crucial for languages with rich morphology, we found generators for agglutinative languages such as Tamil (Kengatharaiyer et al., 2021), Tatar (in Apertium, `https://github.com/apertium/apertium-tat`), and for many uralic languages (Prószéky and Novák, 2005; Novák, 2008a,b; Bakró-Nagy et al., 2010; Endrédy et al., 2010; Fejes and Novák, 2010)[3].

There are some morphological analyzers for Hungarian that can be used for generation as well. HuMor also has a Hungarian morphological generator mode (Novák, 2015) which was the basis for the MetaMorpho machine translation system (Novák et al., 2008) as well. HunSpell can be used for this task in two ways[4]: it generates word forms by typing the lemma and the features, or typing the lemma and an example word (e.g. *kutya* + *macskákkal* = *kutyákkal*). Hunmorph (Trón et al., 2005) and Morphdb.hu (Trón et al., 2006) are also suitable for morphological generation. Hunmorph-foma[5] uses the morphological tagset of HunMorph and it is based on the foma generator (Hulden, 2009). Our goal was to generate word forms using the tagsets of emMorph (Novák et al., 2016; Novák et al., 2017) and Universal Dependencies, therefore the generators referred here are not suitable for our needs, because they use other tagsets.

In contrast to the solutions described above, neural networks can also be used to solve the task of generation. E.g. Malouf (2016) uses a Long Short-Term Memory (LSTM) network to learn the paradigms of a morphologically complex language and the model generates the paradigms of Russian, Finnish, Irish, Maltese, and Khaling. Micher (2019) built a generator from the output of an existing analyzer for Inuktitut with a sequence-to-sequence neural network which transforms the underlying morphemes into the surface forms. Schwartz et al. (2019) created a generator for the case-inflected nouns in the polysynthetic language of Yupic. They utilized an existing FST morphological analyzer to

[1] https://huggingface.co/NYTK/morphological-generator-emmorph-mt5-hungarian

[2] https://huggingface.co/NYTK/morphological-generator-ud-mt5-hungarian

[3] The morphological analyser and word form generators for the languages mentioned in these papers can be found and used at `http://www.morphologic.hu/urali/index.php?lang=english`.

[4] See the methods of pyhunspell, the Python bindings for HunSpell here: `https://github.com/pyhunspell/pyhunspell/wiki/Documentation`

[5] `https://github.com/r0ller/hunmorph-foma`

create training data and treated morphological generation as a recurrent neural sequence-to-sequence task.

3 Corpora

For training neural models for the task of morphological generation, as a first step, a huge amount of morphologically analysed text was needed. Since Webcorpus 2.0 (Nemeskey, 2020) is a morphologically analysed corpus of 9 billion tokens, it seemed the right choice for this task. In the morphological generation process, using the lemma and its' morphological tags of a word, the model generates the correct surface form. Thus, from the analysed Webcorpus 2.0 the following columns are needed: FORM (original surface form), LEMMA (lemma), XPOS (emMorph tags). From these extracted columns, we created our emMorph corpus. In this corpus, one segment is one word and its forms: *munka* `[/N][Acc]` *munkát*.

Our first experiment created two versions of this corpus: **unnormalised** and **normalised**. In the case of the unnormalised corpus, we simply applied the `uniq` function, so every $lemma + tag = form$ segment was represented only once. Using this method, 75,569,032 segments (and type at the same time) were created. But after the training and evaluation processes, many errors were recognized. For instance, there were many words without accents in this corpus. After `uniq` process, for example, the accusative form of *kutya* 'dog' can be *kutyat* and *kutyát* with the same probability. In this case, *kutyat* is the wrong form. Our models with the input `kutya [/N][Acc]` generated the *kutyat* word form, which is incorrect. Thus, we needed to normalise our corpus. Our normalization steps are the following:

- The frequency of the $lemma + tag = form$ segments were calculated based on the Webcorpus 2.0
- A hard rule was used which filters out the less frequent segments with the same $lemma + tag$. This constrain had more than 95% precision to select wrong segments and there were only few cases where it removes correct word forms.
- The segments that has a final subtag `[Punct]` were filtered out. We decided to add this step as the generator shouldn't know which punctuation mark should be generated at the end of a word.
- The rare segments (less then 10 times presented in the corpora) were removed.
- The word frequency information would be integrated to the model so all segments were duplicated based on the logarithm of their word frequency.

After the normalization process, our corpus contained 12,371,157 segments and 6,830,804 types.

In our next step, we created our UD corpus. The Webcorpus 2.0 does not contain UD tags. Thus, we needed an emMorph→UD converter. For this task, we used emmorph2ud2 (Indig et al., 2019) converter[6].

[6] `https://github.com/vadno/emmorph2ud2`

4 Methods and Experiments

For the training of the morphology generator, we applied two approaches: machine translation and text generation.

Machine translation: Our first approach is solving the task as a translation task. Technically they are a sequence-to-sequence architectures, which contains not only a generator, but an encoder part as well. The source segment contains the lemma and the morphological tags, the target segment is the morphologically analysed surface word. An emMorph and a UD example are shown in Table 1.

Tagset	Source format	Target format
emMorph	munka [/N][Acc]	munkát
UD	munka NOUN Case=Acc|Number=Sing	munkát

Table 1. A sample of the format of the data

To train machine translation models, three different kinds of method were tried:

- **Marian NMT**: Marian NMT is a machine translation framework, which has written in C++ language. The biggest advantages of this seq2seq implementation are the optimized training and prediction time, as well as the low resource and hardware requirement. We trained models from scratch with the following hyperparameters: epoch: 10, dim-vocabs 800, learn-rate 0.00005, max-length 50, dim-emb 512. The system was trained on a single NVIDIA A100 (20GB) GPU. The training took 12 hours.
- **mT5**: We fine-tuned the pre-trained google/mt5-base model to the translation task. The hyper-parameters are the following: source prefix: 'morph:'; max source length: 64; max target length: 32; batch size: 128/GPU (8 GPU); source language: en; target language: hu; epoch: 10. The training took 14 hours.
- **M2M100**: Since this task was considered a translation task, we fine-tuned the facebook/m2m100_1.2B model that was pre-trained for multilingual machine translation tasks. The hyper-parameters are the following: max source length: 64; max target length: 32; batch size: 128/GPU (8 GPU); source language: en; target language: hu; epoch: 10. The training took 15 hours.

Text generation: Our second approach is solving the task as a text generation task. It means that one segment contains all the information: the lemma, the morphological tags and the morphological analysed surface word. To help the generative model to solve the task more precisely, separator and end-of-text tags were added to the segment. An emMorph and an UD example are shown in example 1.

(1) EM: munka [/N][Acc] </s> munkát <|endoftext|>
UD: munka NOUN Case=Acc|Number=Sing </s> munkát <|endoftext|>

To train the text generation model, a GPT-2 model was fine-tuned:

– **GPT-2**: We have fine-tuned the PULI GPT-2 model to this task. The hyperparameters are the following: block size: 64; batch size: 128/GPU (4 GPU); epoch: 10. In the fine-tuning script, we have modified the preprocess function. In the original function, the input texts were concatenated. We removed this concatenation method. As a consequence, one segment contains only one word and its forms. In our experiments we tried two kinds of separator tags: '</s>' and '='. During the measurements, using the '=' separator tag, we could gain higher results. Thus, in the results section, we presented the performance of models that used the '=' separator. The training took 8 hours.

The first experiment was to train the model on the 'raw' version of our corpora. It means the models were trained on the unnormalised corpus. The Marian on this corpus achieved 94.5% word-based accuracy, but on the NerKor corpus the Marian model could gain only 76.3% accuracy. This experiment showed that the Webcorpus contains many erroneous word forms. Thus, our next step was normalizing the corpus, then the models were retrained on the normalised corpus.

All of our models were trained on NVIDIA A100 (80GB) GPUs.

5 Results and Evaluation

To evaluate our models, we used the word-base accuracy metrics. Table 2 shows the results of our models on the morphologically analysed subpart of NerKor. In general, all models could achieve more than 93% accuracy, which means all of our models could learn this task. Among these models, Marian could achieve the highest performance. This indicates that in this task the pre-training process could not add any extra knowledge. The surprising result is that the GPT-2 gained the lowest values. In our hypothesis, this task fits the text generation task.

	emMorph (%)	UD (%)
Marian real accuracy	**99.43**	**98.69**
Marian	**96.27**	**94.94**
mT5	95.53	94.66
M2M100	95.04	93.83
GPT2	93.78	93.43

Table 2. Results

Since Marian achieves the best results, our evaluation processes were applied to the output of the Marian models.

5.1 The evaluation of the emMorph-based generation

For the evaluation, the morphologically analyzed subpart of NerKor was used. The test cases are the tokens generated based on the lemmas and the disambiguated emMorph tags from the corpus. These generated tokens were compared to their counterparts found in the corpus. The generated token was identical to the token in the corpus in 53,942 cases. The remaining 2,006 tokens were checked manually to explore the reasons for the difference between the generated tokens and the ones in the corpus.

- *erroneous reference*: It turned out that in some cases (actually, in 1,317 cases) the generator was not at fault. On the one hand, despite NerKor being a gold standard corpus in which the morphological tags were checked and corrected by two annotators and were curated by a third, some annotation errors may naturally occur in the corpus.[7] On the other hand, the annotation scheme of NerKor caused some different tokens compared to the generated ones, especially in the cases of named entities. Due to the annotation scheme of NerKor the morphological tags of the named entities do not reflect the internal structure of the tokens of named entities. Only the last token of multiword named entities got the full morphological tag, all other tokens got the noun tag without case suffix (`[/N]`).
- *allomorphy*: In 382 cases the token in the corpus and the generated one were also correct (e.g. *estig, estéig*, *panelban, panelben*, *tietek, tiétek*).
- *actual errors*: only 306 tokens turned out to be a real mistake made by the generator.

5.2 The evaluation of the UD-based generation

Of the 55,282 tokens in the test set of NerKor, 52,484 tokens were correctly generated by the model. The 2,798 instances, where there was a difference between the original word form and the model's output, were manually checked. Three cases are differentiated:

- *erroneous reference*: 1,728 instances are extracted from our evaluation as the reference morphological features (*használjuk*, lemma: `használ`, features: `Definite=Ind|Mood=Imp|Number=Plur|Person=1|Tense=Pres|VerbForm=Fin|Voice=Act`)[8] or the lemma (*írt* 'wrote', lemma: `írt`, features: `Definite=Ind|Mood=Ind|Number=Sing|Person=3|Tense=Past|VerbForm=Fin|Voice=Act`) are erroneous, or the reference word form contains punctuation marks as the result of false tokenisation (*használatát.*), or the output of the generator is correct, but the reference word form is not normalised (*károsító*).

[7] A useful by-product of the error analysis is the list, based on which the incorrect tags of NerKor can be corrected.

[8] The false tag in the feature set is highlighted with red.

– *allomorphy*: In 349 cases both the reference word form and the generated word form are correct. This is mainly caused by the allomorphs in Hungarian, such as *azzal* = *avval*. As the two essive suffixes, *essivus-modalis* and *essivus-formalis* are uniformly tagged as `CASE=ESS`, the word forms ending in *-ként* and *-ul/-ül* are not differentiated. Parallelly, the two types of plural suffices (plural *kutyá**k*** 'dogs' and familiar plural *Nagyé**k*** 'the Nagys') are not differentiated either. These are the other main cause of this type of error in the test set.
– *actual errors*: In 720 cases the reference was correct, and the model generated a false word form. This category includes tokens like emojis, foreign segments and various punctuation marks.

The biggest challenge for the generator seems to be the case of numerals. In the test set, many different surface forms bear the exact same lemma and morphological features, see for example Table 3. Since Universal Morphology (McCarthy et al., 2020) allows a more subtle distinction between different types of numerals,[9] it would be useful, not only for the morphological generator but also for other models fine-tuned on this corpus, to refine these.

reference	lemma	POS	features	generated
sok	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
legöbb	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
legtöbb	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
legtöbben	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
sokan	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
sokszor	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
tobb	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
több	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
többen	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
többször	sok	NUM	Case=Nom|Number=Sing|NumType=Card	legtöbbször
többre	sok	NUM	Case=Sub|Number=Sing|NumType=Card	legtöbbre
legtöbbön	sok	NUM	Case=Sup|Number=Sing|NumType=Card	többön

Table 3. A sample of numerals in the test set of NerKor. *Reference* column shows the word forms in the corpus; *lemma*, *POS* and *features* are the analysis of the word form in the corpus; *generated* column shows the word forms generated by the model based on the *lemma*, *POS* and *features*.

In Table 4, you can see the error types of the Marian models. In both cases more than 60% of errors are reference errors and more than 10% of errors are allomorphy, which are actually correct. Thus, according to the error ratios, our Marian emMorph model could achieve **99,43%** and Marian UD model could gain **98,69%** real accuracy (see Table 2).

[9] `https://universaldependencies.org/u/pos/NUM.html`

Error Type	Ratio (%)	Sample lemma (`tag`) – reference – generated
		emMorph
Reference	65.70	tápoldat (`[/N][Acc]`) – tépoldatot – tápoldatot
Allomorphy	19.04	panel (`[/N][Ine]`) – panelben – panelban
Model error	**15.28**	te (`[/N|Pro][All][2Sg]`) – hozzád – tédhez
		UD
Reference	61.78	vég (`Case=Nom`)[10] – véget – vége
Allomorphy	12.47	agresszív (`Case=Ess`) – agresszívan – agresszíven
Model error	**25.73**	végtelen (`Pos=NUM`) – végtelen – végtelen-

Table 4. Error types of the Marian models

6 Summary

Here we presented a morphological generator for Hungarian that generates surface forms from emMorph and UD morphological tags with high accuracy. We tackled the problem with two approaches: first, neural machine translation models were trained based on the morphological analysis as the source format and the corresponding surface form as the target format. Second, we experienced with text generation, where the morphological analysis is followed by the correct word form. The corpus we used is a normalised version of Webcorpus 2.0. From the above-mentioned methods, Marian MT proved to produce the best results: 96.27% in the case of emMorph and 94.94% in the case of UD. We evaluated its output manually on NerKor. Our detailed analysis shows that the generator achieves a high accuracy of 99.43% (emMorph) and 98.69% (UD). This model may be used for several NLP tasks, such as anonymisation and terminology translation. Our model called *HuMorGen* is freely available on GitHub.[11]

Bibliography

Bakró-Nagy, M., Endrédy, I., Fejes, L., Novák, A., Oszkó, B., Prószéky, G., Szeverényi, S., Várnai, Z., Wagner-Nagy, B.: Online morfológiai elemzők és szóalak-generátorok kisebb uráli nyelvekhez. In: Attila, T., Veronika, V. (eds.) VII. Magyar Számítógépes Nyelvészeti Konferencia: MSZNY 2010. pp. 345–349 (2010)

Endrédy, I., Fejes, L., Novák, A., Oszkó, B., Prószéky, G., Szeverényi, S., Várnai, Zs., Wágner-Nagy, B.: Nganasan - Computational Resources of a Language on the Verge of Extinction. In: Sarasola, K., Tyers, F.M., Forcada, M.L. (eds.) Creation and Use of Basic Lexical Resources for Less-Resourced Languages: 7th SaLTMiL Workshop (LREC-2010). pp. 41–44 (2010)

[11] Link in the final version of the paper.

Fejes, L., Novák, A.: Obi-ugor morfológiai elemzők és korpuszok. In: Attila, T., Veronika, V. (eds.) VII. Magyar Számítógépes Nyelvészeti Konferencia: MSZNY 2010. pp. 284–291 (2010)

Hulden, M.: Foma: a Finite-State Compiler and Library. In: Proceedings of the Demonstrations Session at EACL 2009. pp. 29–32. Association for Computational Linguistics, Athens, Greece (Apr 2009), https://aclanthology.org/E09-2008

Indig, B., Sass, B., Simon, E., Mittelholcz, I., Vadász, N., Makrai, M.: One format to rule them all – The emtsv pipeline for Hungarian. In: Proceedings of the 13th Linguistic Annotation Workshop. pp. 155–165. Association for Computational Linguistics, Florence, Italy (2019)

Kengatharaiyer, S., Dias, G., Butt, M.: ThamizhiMorph: A morphological parser for the Tamil language. Machine Translation 35, 1–34 (04 2021)

Malouf, R.: Generating morphological paradigms with a recurrent neural network. San Diego Linguistics Papers 6, 122–129 (09 2016)

McCarthy, A.D., Kirov, C., Grella, M., Nidhi, A., Xia, P., Gorman, K., Vylomova, E., Mielke, S.J., Nicolai, G., Silfverberg, M., Arkhangelskiy, T., Krizhanovsky, N., Krizhanovsky, A., Klyachko, E., Sorokin, A., Mansfield, J., Ernštreits, V., Pinter, Y., Jacobs, C.L., Cotterell, R., Hulden, M., Yarowsky, D.: UniMorph 3.0: Universal Morphology. In: Proceedings of the Twelfth Language Resources and Evaluation Conference. pp. 3922–3931. European Language Resources Association, Marseille, France (May 2020), https://aclanthology.org/2020.lrec-1.483

Micher, J.: Bootstrapping a Neural Morphological Generator from Morphological Analyzer Output for Inuktitut (2019)

Minnen, G., Carroll, J., Pearce, D.: Robust, applied morphological generation. In: Proceedings of the First International Conference on Natural Language Generation - Volume 14. p. 201–208. INLG '00, Association for Computational Linguistics, USA (2000), https://doi.org/10.3115/1118253.1118281

Nemeskey, D.M.: Natural Language Processing Methods for Language Modeling. Ph.D. thesis, Eötvös Loránd University (2020)

Novák, A., Siklósi, B., Oravecz, Cs.: A New Integrated Open-source Morphological Analyzer for Hungarian. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). pp. 1315–1322. European Language Resources Association (ELRA), Portorož, Slovenia (may 2016), https://aclanthology.org/L16-1209

Novák, A.: Creating a Morphological Analyzer and Generator for the Komi language. In: Carson-Berndsen, J. (ed.) Proceedings of the SALTMIL Workshop at LREC 2004. pp. 64–67 (2008a)

Novák, A.: Language resources for Uralic minority languages. In: Williams, B., Forcada, M.L., Sarasola, K. (eds.) Proceedings of the SALTMIL Workshop at LREC 2008: Collaboration: interoperability between people in the creation of language resources for less-resourced languages. pp. 27–32 (2008b)

Novák, A.: A Model of Computational Morphology and its Application to Uralic Languages. Ph.D. thesis (2015)

Novák, A., Rebrus, P., Ludányi, Zs.: Az emMorph morfológiai elemző annotációs formalizmusa [Annotation format of the emMorph morphological analyzer]. In: XIII. Magyar Számítógépes Nyelvészeti Konferencia. pp. 70–78. Szegedi Tudományegyetem, Szeged (2017)

Novák, A., Tihanyi, L., Prószéky, G.: The MetaMorpho Translation System. In: Proceedings of the Third Workshop on Statistical Machine Translation. pp. 111–114. Association for Computational Linguistics, Columbus, Ohio (2008)

Prószéky, G., Novák, A.: Computational Morphologies for Small Uralic Languages. In: Antti, A., Carlson, L., Linden, K., Piitulainen, J., Suominen, M., Vainio, M., Westerlund, H., Yli-Jyrä, A. (eds.) Inquiries into Word, Constraints and Contexts. (Festschrift in the Honour of Kimmo Koskenniemi on his 60th Birthday), pp. 150–157 (2005)

Schwartz, L., Chen, E., Hunt, B., Schreiner, S.L.: Bootstrapping a neural morphological analyzer for St. Lawrence Island Yupik from a finite-state transducer. In: Proceedings of the 3rd Workshop on the Use of Computational Methods in the Study of Endangered Languages Volume 1 (Papers). pp. 87–96. Association for Computational Linguistics, Honolulu (Feb 2019), https://aclanthology.org/W19-6012

Simon, E., Vadász, N.: Introducing NYTK-NerKor, A Gold Standard Hungarian Named Entity Annotated Corpus. In: Ekstein, K., Pártl, F., Konopík, M. (eds.) Text, Speech, and Dialogue - 24th International Conference, TSD 2021, Olomouc, Czech Republic, September 6-9, 2021, Proceedings. Lecture Notes in Computer Science, vol. 12848, pp. 222–234. Springer (2021)

Trón, V., Gyepesi, G., Halácsy, P., Kornai, A., Németh, L., Varga, D.: Hunmorph: open source word analysis. In: Jansche, M. (ed.) Proceedings of the ACL 2005 Software Workshop, pp. 77–85. ACL, Ann Arbor (2005)

Trón, V., Halácsy, P., Rebrus, P., Rung, A., Vajda, P., Simon, E.: Morphdb.hu: Hungarian lexical database and morphological grammar. In: Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06). European Language Resources Association (ELRA), Genoa, Italy (May 2006), http://www.lrec-conf.org/proceedings/lrec2006/pdf/683_pdf.pdf