# Glyph embedding for neural machine translation

Kuan Yu
kuanyu@uni-potsdam.de

Master's Program in *Cognitive Systems*
University of Potsdam

February 2019

## Abstract

We explored the possibility of using glyph images as character embedding for neural machine translation with a simple recurrent attentional encoder-decoder architecture. We found that as the input to the decoder, glyph embedding worked as effectively as learned character embedding, even when the writing system was not logographic. We also tried glyph prediction and successfully performed end-to-end language translation on the written form. In this paper we analyzed the successes and failures of our models to summarize the advantages of our glyph-based approach in comparison to the conventional text-based approach, and proposed exciting possibilities for future works.

## 1 Introduction

Language stored in text format on computers rely on a standard of character encoding, where each character is assigned a code point. Sequences of characters, namely strings, are generated from the set of characters through the free monoid construction, also known as the Kleene closure in computer science. This method of encoding has been the basis of natural language processing. Expert systems model language at the symbolic level. Symbols are strings representing words or phrases, which are treated as discrete units and used for constructing larger units of language, such as sentences, paragraphs, or whole texts. The relations between symbols are stored externally as logical rules. With the successful and wide adoption of statistical models, especially deep neural networks, symbols require an additional layer of encoding. Statistical methods typically pick a minimal level of units to treat as discrete values. Common choices for the modeling level are words, ngrams, or characters. The starting point is then to embed the set of minimal units in a vector space as the standard basis. This is known as one-hot encoding. Often a linear transformation is applied to the one-hot vectors, to model the linguistic relations between these units by their geometric relations. This learned linear transformation is called a learned embedding. Larger units are then composed from the vector representation of these minimal units through algebraic operations. In this work, we adopt a nostalgic approach based on writing systems. We use glyph images as character embedding in the task of neural machine translation (NMT).

NMT systems typically translate sentence strings from a source language to sentence strings in a target language. When a format other than text is required, such as speech, it is common to compose the text translation system with a speech recognition and/or synthesis system in a pipeline (Dureja and Gautam 2015). End-to-end speech translation, which usually refers to speech-to-text translation, is an active field of research (Bérard et al. 2018; Vila et al. 2018), while research into direct speech-to-speech translation is uncommon (Serdyuk et al. 2018). To our knowledge, this is the first attempt at end-to-end written language translation. On top of showing that translation directly on the written form is possible and just as effective as text translation under certain conditions, we demonstrate the advantages of our approach over the conventional text-based approach with a careful examination of the successes and failures of our glyph-based models.

In this paper we first describe the background in NMT (Section 2), then explain our experiments (Section 3), and finally describe our results (Section 4) and present our conclusion (Section 5). An open-source implementation for our experiments is available in Python3 using Tensorflow.[1]

[1]https://github.com/lepidodendron/lepidodendron
This project is a joint-effort of the author and Bailey Polonsky at the University of Potsdam.

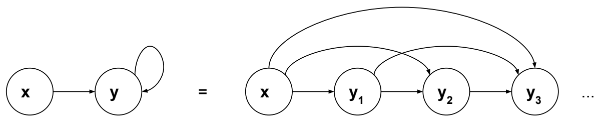


Figure 1: An autoregressive encoder-decoder models the probability of a target sequence $y$ given a source sequence $x$, with the target sequence factorized causally by position: $p(y \mid x) = p(y_1 \mid x)\ p(y_2 \mid y_1, x)\ p(y_3 \mid y_2, y_1, x)\ \ldots$

## 2 Background

### 2.1 Neural machine translation

Recurrent neural network (RNN) with an encoder-decoder structure has led to the success of NMT systems (Kalchbrenner and Blunsom 2013; Sutskever, Vinyals, and Le 2014; Cho, Van Merriënboer, Gulcehre, et al. 2014; Cho, Van Merriënboer, Bahdanau, et al. 2014; Wu et al. 2016). The encoder RNN receives a sequence of vectors from the source language, typically a sentence encoded on the word level, and produces a single state vector which encodes the sentence. The decoder autoregressively generates a sequence in the target language, with each step conditioning on the encoded state and previously generated partial sequence. The conditioning is conducted by the decoder recurrent state, initialized from the encoded state. Figure 1 shows the corresponding graphical model.

During inference, an output is sampled from the decoder predictions at each step, and fed back together with the recurrent state for the next step. Since the search space grows exponentially for such a structure prediction problem, beam search is commonly used to approximate the optimal solution (Freitag and Al-Onaizan 2017). A cheaper alternative is greedy decoding, where the sampling is simply performed by picking the prediction with the highest probability. For training an autoregressive model, the teacher-forcing method is typically used (Williams and Zipser 1989). Instead of running the feedback loop, the true target sequence is given as input with a begin-of-sentence (BOS) symbol padded at the front, to predict the same sequence with an end-of-sentence (EOS) symbol padded at the end.

More recent works favor various attention mechanisms to counter the drawback of having to encode a whole sequence into a single state vector (Bahdanau, Cho, and Bengio 2014; Luong, Pham, and Manning 2015). The encoder RNN produces an annotation vector for each position in the source sequence, and for each decoding step, the annotation vectors are summarized into a single context vector using attention according to the recurrent state. The context vector is then combined with the recurrent output in the decoder. A particularly effective attention mechanism is multi-head scaled dot-product attention used in the Transformer architecture (Vaswani et al. 2017).

### 2.2 Modeling level

While words are a common and intuitive choice for the basic units of sentences, it has many drawbacks. The vocabulary size is huge, often in an order of magnitude around or above $10^6$, resulting in a large number of parameters in the embedding matrix. On top of that, due to their power law distribution, the majority of words in the vocabulary of a language remains unseen in the available training data, and most words in the observed vocabulary occur infrequently. Furthermore, morphology is an important part of language, especially for agglutinative languages, which can only be modeled at the subword level. Current approaches often augment word-level modeling with subword units (Sennrich, Haddow, and Birch 2015; Kudo 2018).

Modeling at the character-level can be effective as well (Kalchbrenner, Espeholt, et al. 2016). However, characters also follow a power law or exponential distribution. The Unicode encoding contains over $10^6$ code points with 137 439 defined so far, the vast majority of which never occur in any single corpus. At the same time, character crossover from another language is common. Even on the character-level, it is difficult if not impossible to model a language fully.

Moreover, languages with logographic writing systems encode certain semantic information below the character level. Their compositional structures can only be retrieved from the character glyphs. Learning an embedding from glyphs has shown to be valuable (Su and Lee 2017).

## 3 Experiments

### 3.1 Dataset and preprocessing

We used the European Parliament Proceedings Parallel Corpus for German to English translation (Koehn 2005). The reason for not choosing a logographic language is that we wish to verify the general applicability of our approach.

We kept only parallel instances with sentences containing 3–256 characters. The lower threshold was set to remove trivial instances, and the upper threshold was set for the ease of experimentation. We randomly picked 4 096 instances for validation, and used the remaining 1 574 071 instances for training.

We considered the subset of characters with 99.95% coverage to be the frequent characters. In the source language German, 78 out of 306

(a) German (b) English

Figure 2: Mean of glyphs.

observed characters are frequent, and in the target language English, 72 out of 293 are frequent.

## 3.2 Glyph rendering

For rendering the character glyphs, we used the Pillow[2] library with the Noto Sans Mono[3] font. Each glyph was rendered as an image with pixels in grayscale scaled in range $[0, 1]$.

Individually rendered glyph images vary in size. We fixed the height and width to the maximum value among the frequent characters. With font size 20, the height for German and English glyphs was respectively 25 and 20 pixels. The width was 12 pixels for both. Each rendered image was flattened into a vector. There vectors were the glyph embedding.

We used the whitespace character as BOS paddings and the full block character █ as EOS paddings. We also used the replacement character � for unknown characters, which during training were the infrequent ones. The treatment for unknown characters is only relevant in trials where we used one-hot encoding instead of glyph embedding. When glyphs are used, we always rendered the original character with no special treatment for the rare ones.

Glyph rendering was a severe bottleneck for the training speed. To solve this problem, we cached the frequent glyphs, and only rendered the rare ones on the fly.

[2]https://python-pillow.org/

[3]https://www.google.com/get/noto/

## 3.3 Experiment setup

We conducted four main trials, varying the type of values connected to the model in three locations: the encoder input, the decoder input, and the decoder output. We use `g` to denote when a location is connected to glyphs, and `c` when it is connected to one-hot encoded characters.

When one-hot encoding is used for the input, the linear transformation in the first input layer acts as a learned embedding. Therefore our trials was designed to compare the behaviors of learned embedding and glyph embedding. The four trials were `ccc`, `cgc`, `cgg`, and `ggg`. As an elaboration, in trial `cgc` the encoder used learned embedding while the decoder used glyph embedding but predicted characters instead of glyphs. Trial `ccc` was the baseline in line with conventional text-based translation. Trial `ggg` performed end-to-end translation on the written form.

In order to compute the BLEU scores for evaluating the translation quality, we needed the predictions in text format. Whenever a model was trained to predict the glyphs, we also included an additional output layer for predicting the characters.

## 3.4 Model architecture

For the encoder, we used 3 layers of stacked bidirectional RNN with gated recurrent units (GRU) (Cho, Van Merriënboer, Gulcehre, et al. 2014). The decoder consisted of 3 layers of unidirectional GRU RNN, followed by an 8-head scaled dot-product attention layer (Vaswani et al. 2017) with residual connection (He et al. 2016)

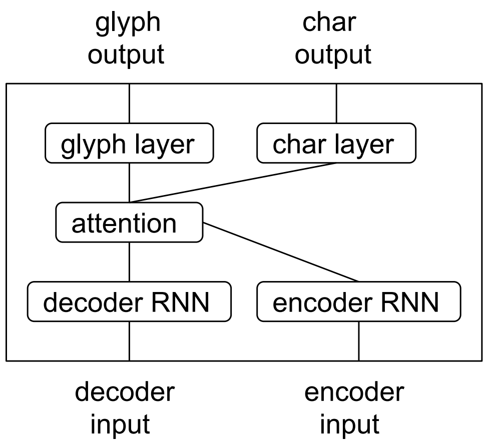


Figure 3: The architecture with both glyph and character predictions.

and layer normalization (Ba, Kiros, and Hinton 2016). When a model was trained to predict characters, we placed a fully-connected layer after attention to predict the logit probabilities over the character set, which included the frequent characters, the proxy unknown character, and the proxy EOS character. When a model was trained to predict glyphs, we placed a fully-connected layer after attention with one unit for each pixel. Figure 3 shows the architecture when both predictions were included.

The loss function for character prediction was softmax cross-entropy (XENT). For glyph prediction, we clipped the outputs in range $[0, 1]$ and computed the mean absolute error (MAE) for the loss. Since the MAE loss was roughly 1/10 of the XENT loss, when both losses were included, we used $10 \times \text{MAE} + \text{XENT}$ as the total weighted loss. During training, we monitored the accuracy of character prediction in teacher-forcing mode as a criterion for fitting and overfitting.

We explored a few alternatives. For the glyph prediction loss, we tried sigmoid cross-entropy, MAE with sigmoid activation, MAE without clipping, and mean squared error. We also tried connecting the character layer after the glyph layer. We found there alternatives less effective.

The models were trained with stochastic gradient descent using Adam optimization (Kingma and Ba 2014). The learning rate was scheduled $lr/(1 + dr \times \sqrt{s})$ by the training step $s$, where $lr = 0.001$ was the initial learning rate and $dr = 0.01$ was the decay rate. We applied 10% dropout (Srivastava et al. 2014) between the RNN layers and after the attention layer right before the residual connection.

We trained the models with mini-batches of 128 instances. Model `ccc` started to overfit after 30 epochs. The other models except for `cgc` did not reach the optimal accuracy, but we stopped training after 60 epochs. Despite being underfit, `cgg` reached a higher validation accuracy than `ccc` and `cgc`, indicating a positive regularization effect when glyphs were used in the decoder.

## 4 Results

### 4.1 Inference modes

During autoregressive inference, the decoder prediction is fed back as the input for the next step. Glyph prediction is a deterministic process, while character prediction is probabilistic, for which we used greedy decoding. A model which was trained to predict both glyphs and characters sometimes disagreed in the two types of predictions (see A.2). We discovered that it worked well to simply match the predicted glyph to the closest character using MAE as a distance measure. This can be seen as a method of discrete quantization: One may consider converting the distance measures to probabilities, making the glyph prediction process probabilistic. We can likewise make character prediction deterministic, by using the probability vector as feedback instead of sampling for a one-hot vector, or by producing a fuzzy glyph averaged according to the predicted probabilities if glyphs are required as input. In total we devised five modes of inference.

1. Use the predicted glyph as feedback and the predicted character for evaluation.
2. Use the predicted glyph as feedback and the closest matching character for evaluation.
3. Match the predicted glyph to the closest character for evaluation and feedback; When glyphs are required as input, render a fresh glyph for the character.
4. Use the predicted character for evaluation and feedback; Render when required.
5. Use the predicted character for evaluation and the probabilities as feedback; Render and average when required.

The inference process terminated when the length of the output sequence reached 256. The predicted sentence was trimmed at the first EOS character.

## 4.2 Results and analysis

We computed the BLEU scores on the validation data using SacreBLEU (Post 2018).[4] Table 1 shows the results. A more detailed analysis with examples can be found in Appendix A.

Mode 1 was the intended usage for the glyph-based models, which suffered from the mismatch between the two types of predictions. Mode 2 avoided this problem. Mode 3 benefited from receiving freshly rendered glyphs as feedback instead of relying on the predicted glyphs which were sometimes fuzzy. Mode 4 conformed to the conventional greedy decoding method and produced the best results. Mode 5 relied on the character prediction but intentionally made the input fuzzy by using the probabilities instead of sampling from them. The results from mode 5 shows that the conventional `ccc` model suffered from the uncertainty introduced by forcing a probabilistic process to be deterministic, whereas the glyph-based models were more robust (see A.7).

| mode | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| `ccc` | n/a | n/a | n/a | 30.9 | 20.3 |
| `cgc` | n/a | n/a | n/a | 30.9 | 22.4 |
| `cgg` | 22.9 | 23.5 | 26.5 | 30.9 | 24.1 |
| `ggg` | 21.9 | 22.2 | 25.3 | 30.2 | 23.0 |

Table 1: BLEU scores.

The two modes which treated translation as a probabilistic process (mode 3&4) produced better results than the deterministic modes. This was due to the nature of the task. During translation, the model often faced multiple possibilities in structuring and wording (see A.1). A probabilistic treatment allowed the model to trim the hypothesis space along the autoregressive process and settled on one eventuality. Since glyph prediction made the process deterministic, the modes which relied on glyphs (mode 1&2&3) produces suboptimal results. However, model `cgc` and `cgg` were as good as the conventional `ccc` when the character prediction was consulted (mode 4). One major advantage of using the glyph prediction was that even rare characters could be predicted, and did not have to be treated as unknowns (see A.6).

The fully glyph-based `ggg` performed worse than `cgg`. The encoder was less effective when

[4] With the `intl` tokenizer (`sacrebleu -tok intl`).

glyphs were used as the input. The Latin script used for the source language provided no advantage to the encoder.

### 4.3 Miscellaneous results

We conducted a follow-up experiment for Chinese to English translation using the United Nations Parallel Corpus (Ziemski, Junczys-Dowmunt, and Pouliquen 2016). We set the upper threshold for the sentence length to 128, resulting in 7 392 227 training instances, and tested with 4 096 instances. The Chinese glyphs were fixed to height 25 and width 20, and 2 681 out of 6 357 character were frequent. We trained `cgg` and `ggg` with mini-batches of size 256 for 12.8 epochs. In mode 2, `ggg` performed slightly better than `cgg` (32.3 vs 31.6 BLEU). However `cgg` performed better in a few other modes. The effectiveness of the encoder when using glyphs for a logographic language was inconclusive. The architecture of our models was not suitable for extracting subglyph structures such as the composition of radicals.

We also experimented with the Transformer architecture (Vaswani et al. 2017) which used self-attention instead of recurrent layers. Since the complexity of self-attention is quadratic with respect to the sequence length, we had to restrict the model to only 2 layers and 256 hidden dimensions in order to match the memory usage and training speed of our RNN models. The reduced model was not able to learn glyph-based translation.

## 5 Conclusion

In this work we explored the possibility of using glyph images as character embedding for machine translation and successfully performed end-to-end language translation on the written form with a simple recurrent attentional encoder-decoder architecture.

With probabilistic inference, glyph embedding in the decoder works as effectively as learned embedding for characters, even when the writing system is not logographic. One major advantage with glyph embedding is that rare characters do not have to be treated as unknowns, and the entire character set can be modeled fully.

With deterministic inference, glyph-based models are more robust in the presence of fuzzy predictions in comparison to conventional text-based models. This is valuable when we wish to avoid the search space for fast translations. Without treating each decoding step as a random variable, the model is not truly autoregressive, which is supported by the fact that fuzzy predictions could occur very early without affecting the decisions regarding the structuring and the wording of the rest of the sentence (see A.1, A.3, A.7). Autoregressive models tend to behave well at the beginning and degenerate as mistakes accumulate.

The pseudo autoregressive behavior of our glyph-based models suggests that glyph embedding and glyph prediction may yield more benefits in a non-autoregressive design (Gu et al. 2017), using techniques such as denoising and iterative refinement (Lee, Mansimov, and Cho 2018), introducing latent variables (Kaiser et al. 2018), and probability density distillation (Oord et al. 2017).

For future works along the glyph-based approach, one may consider using a convolutional network to learn the subglyph composition for a logographic language. Transposed convolution may be used for non-autoregressive decoding. Further more, each sentence may be treated simply as a continuous image, without being segmented into monospaced glyphs. This allows for efficient training using other font types and even handwritten data. For producing the output in text format, one may consider a postprocessing network for character or word predictions with connectionist temporal classification (Graves et al. 2006; Libovický and Helcl 2018).

## References

Williams, Ronald J and David Zipser (1989). "A learning algorithm for continually running fully recurrent neural networks". In: *Neural computation* 1.2, pp. 270–280.

Koehn, Philipp (2005). "Europarl: A parallel corpus for statistical machine translation". In: *MT summit*. Vol. 5, pp. 79–86.

Graves, Alex et al. (2006). "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks". In: *Proceedings of the 23rd international conference on Machine learning*. ACM, pp. 369–376.

Kalchbrenner, Nal and Phil Blunsom (2013). "Recurrent continuous translation models". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1700–1709.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473*.

Cho, Kyunghyun, Bart Van Merriënboer, Dzmitry Bahdanau, et al. (2014). "On the properties of neural machine translation: Encoder-decoder approaches". In: *arXiv preprint arXiv:1409.1259*.

Cho, Kyunghyun, Bart Van Merriënboer, Caglar Gulcehre, et al. (2014). "Learning phrase representations using RNN encoder-decoder for statistical machine translation". In: *arXiv preprint arXiv:1406.1078*.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Srivastava, Nitish et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting". In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems*, pp. 3104–3112.

Dureja, Mahak and Sumanlata Gautam (Nov. 2015). "Speech-to-Speech Translation: A Review". In: *International Journal of Computer Applications* 129, pp. 28–30. DOI: 10.5120/ijca2015907079.

Luong, Minh-Thang, Hieu Pham, and Christopher D Manning (2015). "Effective approaches to attention-based neural machine translation". In: *arXiv preprint arXiv:1508.04025*.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (2015). "Neural machine translation of rare words with subword units". In: *arXiv preprint arXiv:1508.07909*.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton (2016). "Layer normalization". In: *arXiv preprint arXiv:1607.06450*.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Kalchbrenner, Nal, Lasse Espeholt, et al. (2016). "Neural machine translation in linear time". In: *arXiv preprint arXiv:1610.10099*.

Wu, Yonghui et al. (2016). "Google's neural machine translation system: Bridging the gap between human and machine translation". In: *arXiv preprint arXiv:1609.08144*.

Ziemski, Michael, Marcin Junczys-Dowmunt, and Bruno Pouliquen (2016). "The United Nations Parallel Corpus". In: *Language Resources and Evaluation. In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia.*

Freitag, Markus and Yaser Al-Onaizan (2017). "Beam search strategies for neural machine translation". In: *arXiv preprint arXiv:1702.01806*.

Gu, Jiatao et al. (2017). "Non-autoregressive neural machine translation". In: *arXiv preprint arXiv:1711.02281*.

Oord, Aaron van den et al. (2017). "Parallel wavenet: Fast high-fidelity speech synthesis". In: *arXiv preprint arXiv:1711.10433*.

Su, Tzu-Ray and Hung-Yi Lee (2017). "Learning chinese word representations from glyphs of characters". In: *arXiv preprint arXiv:1708.04755*.

Vaswani, Ashish et al. (2017). "Attention is all you need". In: *Advances in Neural Information Processing Systems*, pp. 5998–6008.

Bérard, Alexandre et al. (2018). "End-to-end automatic speech translation of audiobooks". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 6224–6228.

Kaiser, Łukasz et al. (2018). "Fast decoding in sequence models using discrete latent variables". In: *arXiv preprint arXiv:1803.03382*.

Kudo, Taku (2018). "Subword regularization: Improving neural network translation models with multiple subword candidates". In: *arXiv preprint arXiv:1804.10959*.

Lee, Jason, Elman Mansimov, and Kyunghyun Cho (2018). "Deterministic non-autoregressive neural sequence modeling by iterative refinement". In: *arXiv preprint arXiv:1802.06901*.

Libovický, Jindřich and Jindřich Helcl (2018). "End-to-end non-autoregressive neural machine translation with connectionist temporal classification". In: *arXiv preprint arXiv:1811.04719*.

Post, Matt (2018). "A call for clarity in reporting bleu scores". In: *arXiv preprint arXiv:1804.08771*.

Serdyuk, Dmitriy et al. (2018). "Towards end-to-end spoken language understanding". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 5754–5758.

Vila, Laura-Cross et al. (2018). "End-to-End Speech Translation with the Transformer". In: *Proc. IberSPEECH 2018*, pp. 60–63.

## A Examples

Here we include some examples from the validation data. Each example is an image of 3 or 4 lines.

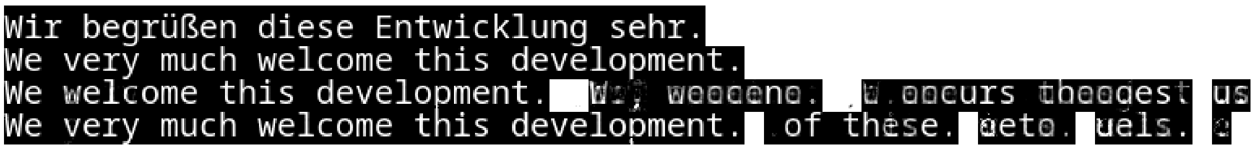
1. The source sentence given as input to the model.
2. The target sentence for reference.
3. The glyph prediction from model cgg.
4. The glyph prediction from model ggg (optional).

Each image is followed by the character predictions. These examples demonstrate the behavior of our models in mode 1. The sentence length is limited to 64 characters.

As a reminder, the full block character █ is used to signal EOS, which appears as a white block in the images. And the replacement character � stands for unknown characters, which only occurs in the character predictions. Many fuzzy glyphs resemble the mean of glyphs in Figure 2.

### A.1 Decision points

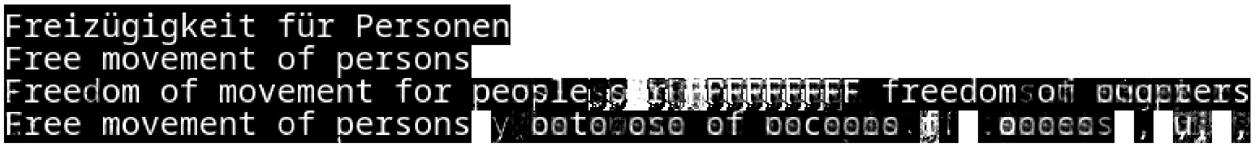
A comparison between the cgg and ggg predictions revealed some choices the model had to face. At those decision points, the glyphs were fuzzy.



Wir begrüßen diese Entwicklung sehr.
We very much welcome this development.
We welcome this development. W; weeeene. .w.encurs theegest us
We very much welcome this development. .of these. aeto. uels.

```
cgg: We velcome this development.█ SS werkine.█ O wncurs toisgest█.s
ggg: We very much welcome this development.█ of these.█ eth █uels.█.█
```

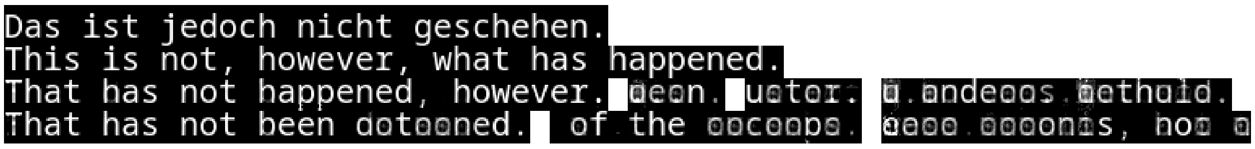
Model cgg was tempted to produce the “v” in very, but simply produced “welcome” instead, resulting in a fuzzy “w” in the glyph prediction and “velcome” in the character prediction.



Freizügigkeit für Personen
Free movement of persons
Freedom of movement for peoples FFFFFFFF freedom of mugpeers
Free movement of persons y boto ose of oocoooo. .ooeees , u ,

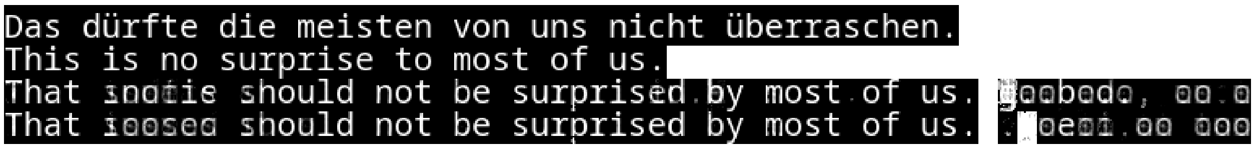
```
cgg: Free om of movement for people████████████free om on migpeers
ggg: Free movement of persons█y brto ovc of tucomaa o.█.ortei █.██(█(
```

“Freedom” vs “Free”; “people” vs “persons”.


Das ist jedoch nicht geschehen.
This is not, however, what has happened.
That has not happened, however. deen. ustor. .andeoos.bethuid.
That has not been dotooned. of the oocoops. oooo.oooonis, hoo

```
cgg: That has not happened.  however.█Nesn.█ustir.█ andenl.  Tuthuld.█
ggg: That has not heen drtammed.█.af the sucomda.█Aask Aitomis.  hot i
```

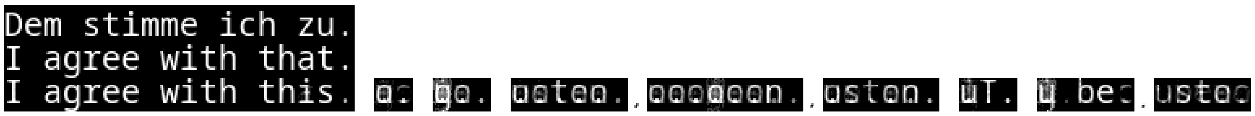
Model ggg mistook the structure for a passive voice construction, and tried to produce a similar structure in English, but failed to find a suitable verb.


Das dürfte die meisten von uns nicht überraschen.
This is no surprise to most of us.
That inotie should not be surprised.by most.of us. oobodo, oo.o
That iooseo should not be surprised by most of us. . oeoi.oo ooo

```
cgg: That itdris should not ce surprised.ay most.of us.█.asbod.  au t
ggg: That stvses should not be surprised by most of us.█.█.er an tun
```

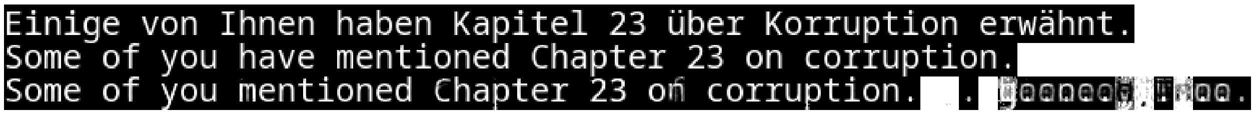
Both model erroneously tried to translate with the passive voice, and failed to find a suitable word to fill the subject position. The failure occurred before the passive construction, indicating that the decision was not made causally.

## A.2 Mismatch between glyph and character predictions


Dem stimme ich zu.
I agree with that.
I agree with this. o. oo. ooteo.,oooooon.,oston. uT. u.be.ustoo

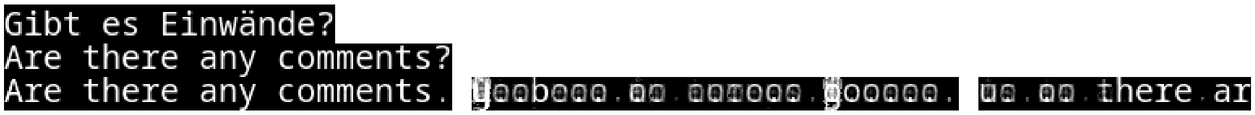
```
cgg: I agree.with thas.█..█.a.█.bt...█........█ust...█.T.█ .bel█.sta.
```

Here the model struggled between “this” and “that”. The glyph prediction settled for “this”, even though it was unsure about the “i”, forcing the character prediction to become “thas”.


Einige von Ihnen haben Kapitel 23 über Korruption erwähnt.
Some of you have mentioned Chapter 23 on corruption.
Some of you mentioned Chapter 23 on corruption. . oooooo..ooo.

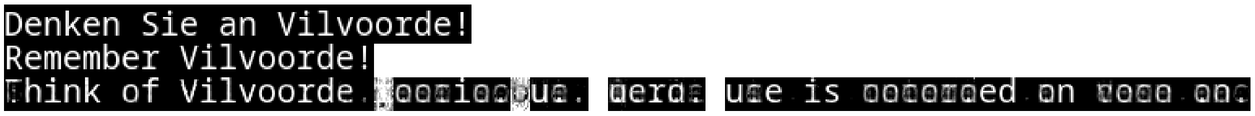
```
cgg: Some of you mentioned Chapter 23 of corruption.█ .█ as.e.  █S█...
```

“on” vs “of”.


Gibt es Einwände?
Are there any comments?
Are there any comments. ooboo.oo.ooooo.oooooo. oo.oo.there.ar

cgg: Are there any comments?█ a?be???an?ture?s??ook???█u???r there?ar

Period vs question mark.



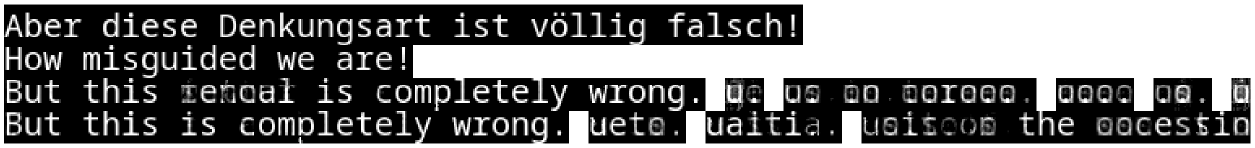
```
Denken Sie an Vilvoorde!
Remember Vilvoorde!
Think of Vilvoorde. [illegible]ua. [illegible] use is [illegible]ed on [illegible]
```

cgg: Think of Vilvoorde�R�aki��█u��█�ar��█use.is tuturaed�an aask��n�

Period vs exclamation mark.

## A.3 Failure in literal translation

When the model tried to produce a literal translation following the structure of the source sentence, it might fail to find a suitable word in the target language.

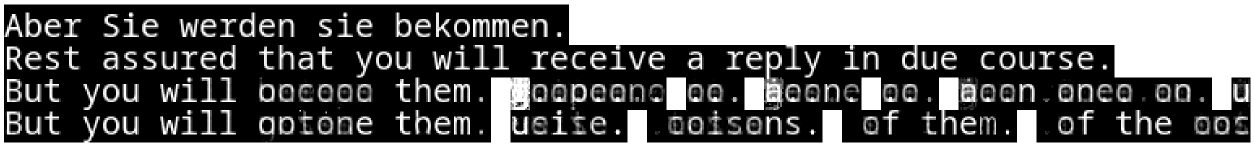


```
Aber diese Denkungsart ist völlig falsch!
How misguided we are!
But this [illegible] is completely wrong. [illegible]
But this is completely wrong. uete. uaitia. usi[illegible] the [illegible]cestin
```

cgg: But this manoul is completely wrong.█��█����n aurr���█����█���█�

ggg: But this ms completely wrong�█ueti�█uasti��█usts p the fttestin

The word “Denkungsart” is difficult to translate. Model cgg failed to find the translation, while ggg avoided it altogether.

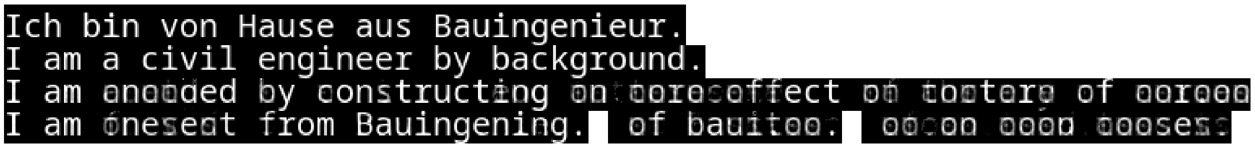


```
Aber Sie werden sie bekommen.
Rest assured that you will receive a reply in due course.
But you will [illegible] them. [illegible]
But you will [illegible] them. ueise. [illegible]sons. of them. of the [illegible]
```

cgg: But you will brcopp them.█...n....█...█.esn.█...█.asn Cner.yu.█.

ggg: But you will gnli e them.█uetle.█.tunlens.█ if them.█.tf the Uuc

Failed to translate “bekommen”.



```
Ich bin von Hause aus Bauingenieur.
I am a civil engineer by background.
I am [illegible]ded by constructing on [illegible]effect on [illegible] of [illegible]
I am [illegible] from Bauingening. of bau[illegible]. [illegible]
```

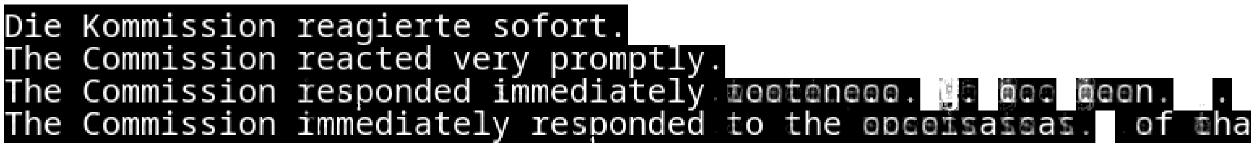
cgg: I am antuied by bonstructing futbarewfffect fn toutiri.bf bur.et

ggg: I am fnerert from Bauingening.█ af Bark..n.█.an tn tans.tensess█

Failed to understand “von Haus aus” and translate “Bauingenieur”.

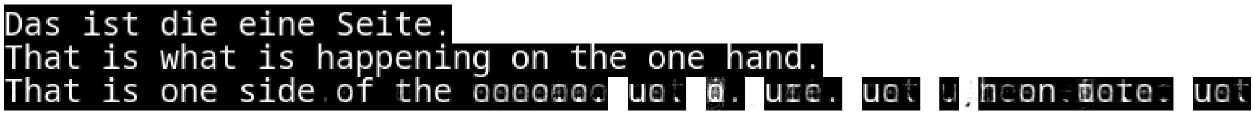
## A.4 Failure in termination

The glyph predictions were conservative about terminating the sentence. Often when the predicted character was a period, the predicted glyph was faint, and the model produced unintelligible content after the point where it should have terminated.



```
Die Kommission reagierte sofort.
The Commission reacted very promptly.
The Commission responded immediately.eootoneoo. [illegible] ooo oeon. .
The Commission immediately responded to the oocoisassas. of tha
```

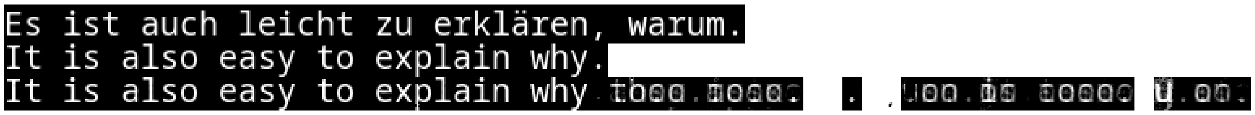
```
cgg: The Commission reaponded immediately.wint ne...██.█...█.ezn.██.█
ggg: The Commission rmmediately responded to the nttenvattat.█.if tha
```



```
Das ist die eine Seite.
That is what is happening on the one hand.
That is one side.of the oooooo uo. o. ure. uot u.hcon.toto. uot
```

```
cgg: That is one side.of the ppc....█.s.█..█urt.█urt█.█rianesrth █ust
```

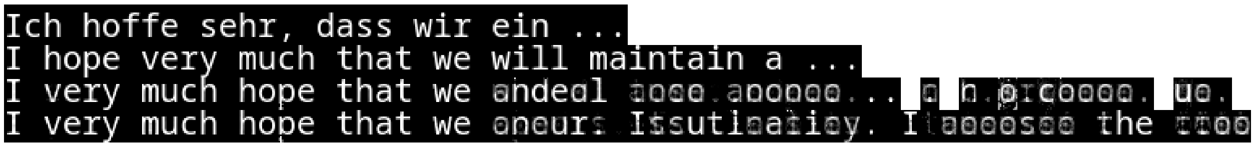
Could have stopped at “one side”.



```
Es ist auch leicht zu erklären, warum.
It is also easy to explain why.
It is also easy to explain why.tooo.iooo. . .uoo.ds ooooo u.oo.
```

```
cgg: It is also easy to explain why.toou.iusk.██.██ in in turt █u an
```

Could have stopped at “why”.

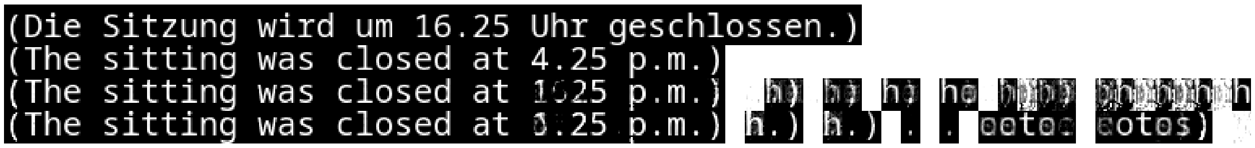


```
Ich hoffe sehr, dass wir ein ...
I hope very much that we will maintain a ...
I very much hope that we andeal oooo.aooooo... o n.orcoooo. uo.
I very much hope that we oneur. Issutioatiey. I aoooso the ttoo
```

```
cgg: I very much hope that we wnderl ..t .n u ....█.█....chrk.  █ e █
ggg: I very much hope that we .neur..�f.u....li....I .lsosee..he....s
```

Both models could not decide how to terminate the sentence due to its incompleteness.

## A.5 Hour format

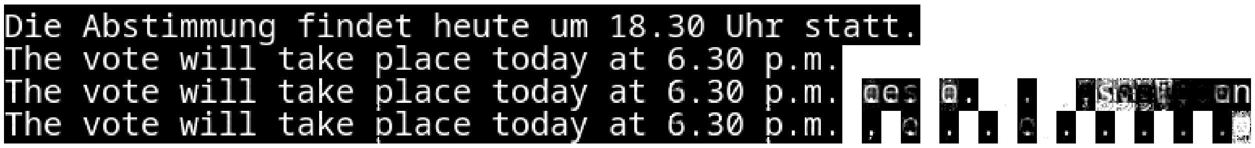


```
(Die Sitzung wird um 16.25 Uhr geschlossen.)
(The sitting was closed at 4.25 p.m.)
(The sitting was closed at 1.25 p.m.)
(The sitting was closed at 3.25 p.m.) h.) h.) . . ootoc otos)
```

cgg: `(The sitting was closed at 4.25 p.m.)██.)█..█h.█h.█D█C.█ChCCChCh`

ggg: `(The sitting was closed at 4.25 p.m.)█h.)█h.)█.█.█.ete-█.otes)██`

Failure.



```
Die Abstimmung findet heute um 18.30 Uhr statt.
The vote will take place today at 6.30 p.m.
The vote will take place today at 6.30 p.m.
The vote will take place today at 6.30 p.m.
```

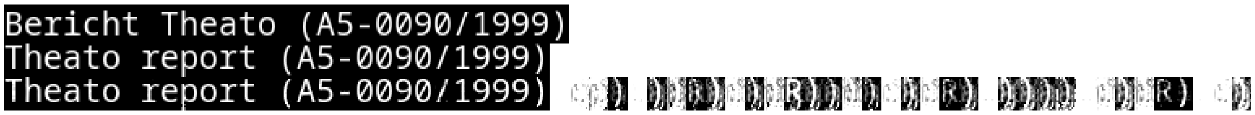
cgg: `The vote will take place today at 6.30 p.m.█.a.█..██.██ █ CRwtn`

ggg: `The vote will take place today at 6.30 p.m.█.█.█.█.█.█.█.█.█.█.(`

Success.

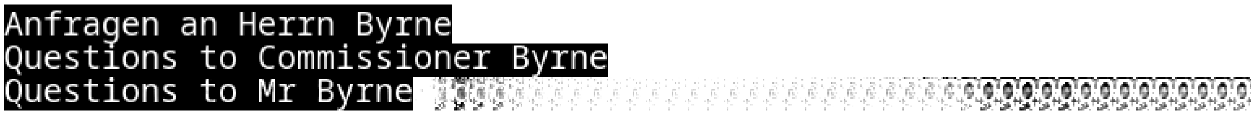
## A.6 Unknown characters

Rare characters could be predicted with the correct glyphs, whereas the character prediction could only produce � since they were treated as unknowns during training.



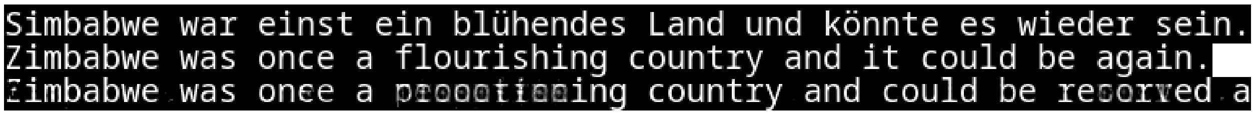
```
Bericht Theato (A5-0090/1999)
Theato report (A5-0090/1999)
Theato report (A5-0090/1999)
```

cgg: `Theato report (A5-0090�1999)██S█ █ █ █ E █ █ █ █ S█ ██ █ )██`



```
Anfragen an Herrn Byrne
Questions to Commissioner Byrne
Questions to Mr Byrne
```

cgg: `�uestions to Cr Byrne█████████████████████████████████████`

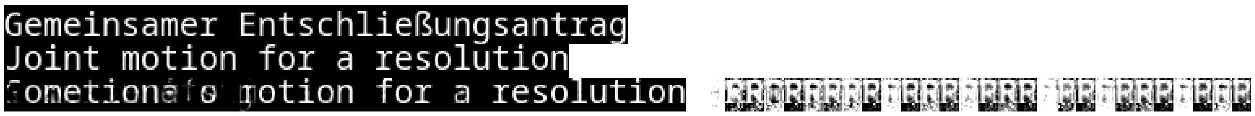


```
Simbabwe war einst ein blühendes Land und könnte es wieder sein.
Zimbabwe was once a flourishing country and it could be again.
Zimbabwe was once a prooetiooing country and could be reoorved a
```

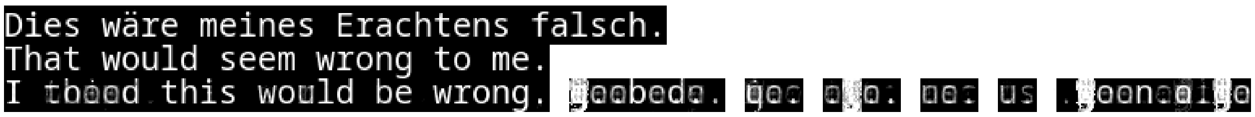
cgg: `�imbabwe was once a pron ri ging country and could be reptrved.a`

## A.7 Robustness against fuzzy predictions

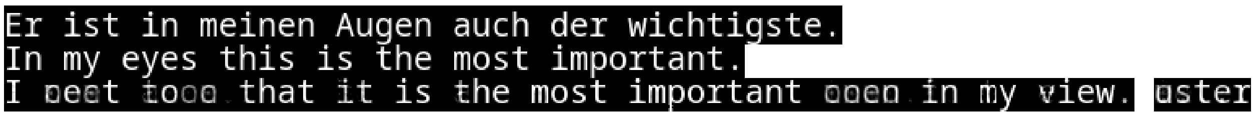
When the model predicted fuzzy glyphs for some words, the rest of the sentence could still follow through, even though the model had never seen a fuzzy glyph during training.


Gemeinsamer Entschließungsantrag
Joint motion for a resolution
Jometiona o motion for a resolution

cgg: Joneriona motion for a resolution████████J██████████████


Dies wäre meines Erachtens falsch.
That would seem wrong to me.
I thood this would be wrong. Jeabeda. Jeo e o. oe. us .Joon e ie

cgg: I tolr this would be wrong.█ asbed..█ e.█ ..█Nat█ s█ (ooniuldr


Er ist in meinen Augen auch der wichtigste.
In my eyes this is the most important.
I oeet tooo that it is the most important ooen in my view. uster

cgg: I tett to k that it is the most important.oucn.in my view.█uster