GENERATING COHERENT TEXT FROM NOISY SPEECH TRANSCRIPTS

Virgile Blanchet-Møhl (s163927), Leif Førland Schill (s153355), and Pranjal Garg (s210242)

ABSTRACT

In this report we present our finetuned noisy speech to coherent text model which was created by finetuning the pretrained Multilingual Lexical Normalization model based on the ByT5 architecture. Our work uses the danish version of the multilexnorm model whice we finetune on our audiobook data to correct noisy speech transcripts generated from audiobooks by an automatic speech recognition model.

Our finetuned model achieves significant improvement in comparison to the pretrained model which demonstrates the significance of applying transfer learning on a pretrained Byte level model for achieving great performance and robustness in the noisy text correction tasks.

The source code is released on githhub and you can access it here: **NoisySpeechCorrectionCode**

Index Terms— Text Correction, ByT5, Finetuning, Transfer learning

1. INTRODUCTION

Automatic speech recognition models are capable of producing accurate transcripts, but these transcripts will often contain a variety of errors. This project aims to remedy this issue through a computationally inexpensive model that takes in noisy transcripts produced by aforementioned models and produces accurate coherent text.

Our model uses the Multilingual Lexical Normalization model[1] which is based on the ByT5[2] model, which we have fine tuned on new data for the Danish language to improve the lexical normalization produced by the multilexnorm for Danish.

2. MATERIALS AND METHODS

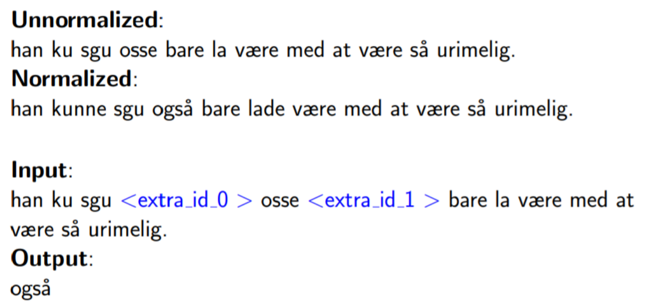
2.1. ByT5

ByT5 is a "token-free" language model, meaning it processes text on the character level. Many other language models require a tokenizer, which assigns a unique ID to each word or subword in the vocabulary. A downside to this method is that out-of-vocabulary strings will be assigned to a special unknown token. This makes these models less robust to noise, and unsuitable for the text correction task since errors in the words are part of the problem. Another benefit of token-free models is that they can freely applied to different languages without needing a new tokenizer.

2.2. MultiLexNorm

This model uses ByT5 as a foundation model in a transfer learning context. It aims to solve the lexical normalization problem, which involves canonizing colloquialisms and informalities as well as removing typos. For the danish variant, the small ByT5 variant (300M parameters) was first trained using large amounts of synthetic data. The synthetic data was produced by introducing typos and common colloquial substitutions to clean text. The model was then fine-tuned on Danish Twitter and Arto text samples. The model uses masking for span of around 20 bytes using special sentinel tokens and reconstructs the masked span. The model achieved best performance in multilingual Lexical Normalization for W-NUT 2021: Multilingual Lexical Normalization shared task.

The model creates a separate input for each word in the sentence, as shown in Figure 2.2 below. The sentinel tokens `<extra_id_0>` and `<extra_id_1>` surround the word that should be corrected.



Unnormalized:
han ku sgu osse bare la være med at være så urimelig.
Normalized:
han kunne sgu også bare lade være med at være så urimelig.

Input:
han ku sgu <extra_id_0 > osse <extra_id_1 > bare la være med at være så urimelig.
Output:
også

Fig. 1. Example of an unnormalized sentence with colloquialisms and the normalized (canonized) reference. The MLN model creates a seperate input for each word with the use of two sentinel tokens.

The model requires the input and reference to be aligned. This process is done so that the model can handle word splits and merges. An example of such an alignment is shown below in Table 1 below

jeg	jeg
købt	købte
ni	niogtredive
og	
tredive	
æbler	æbler
til	til
migselv	mig selv

Table 1. Alignment of input and reference for use in MLN model.

2.3. Evaluation metrics

Two error metrics that can be used for the evaluation of lexical normalization models are the Word Error Rate (WER) and the Error Reduction Rate (ERR).

The WER is related to Levenshtein distance, which is the minimum number of substitutions, insertions, or deletions required to change the input to the reference.

$$\text{LEV} = S + D + I$$

The WER is the Levenshtein distance normalized by the number of words N in the reference.

$$\text{WER} = \frac{S + D + I}{N}$$

The ERR is related to a confusion matrix of the prediction results, where each word in the input is counted as a TN, FP, TP, or FN.

$$\text{ERR} = \frac{TP - FP}{TP + FN}$$

- TN: Annotator did not normalize, system did not normalize.
- FP: Annotator did not normalize, system normalized.
- FN = Annotator normalized, system normalized incorrectly or did not normalize at all.
- TP = Annotator normalized, system normalized correctly.

This metric increases when the model makes correct changes, and decreases when the model changes things that were already correct (despite its name it is not an error measure). It is normalized by the true number of corrections on the input.

2.4. Dataset

The dataset used in this project consists of noisy danish speech transcripts produced from audiobooks, the text of the audiobook being used as a reference. Each data sample contains the transcription of a sentence, the reference text for said sentence, the audiobook of origin and the Word error rate (WER) over the sentence. The WER is used to determine which sentences contain noise and need to be corrected. The data samples from these transcriptions are saved as individual sentences in which the words are mapped into singular strings, thus retaining the context of the sentence when evaluating each individual word.

2.5. Preprocessing

First, the data was aligned as illustrated in Table 1.

The next step was removing examples with extremely high word error rates. This was done because certain inputs were clearly unaligned with their correct reference sentences, that is, the input and reference contained completely different material due to errors in the dataset.

The last step was removing examples with word error rates of zero. Doing this meant all remaining sentences in the dataset had at least one correction to be made. This step affects the specificity of the model after it is trained. We removed the sentences with wer= 0 as we wanted to focus on noisy speech and train the model faster by reducing the size of the dataset by removing unnecessary data points.

2.6. Baseline Models

We used two baseline models: Leave-As-Is (LAI), where nothing in the input was changed, and the MLN(MultilLexNorm) model before it was fine-tuned on our data.

2.7. Training

We used randomly 10% of the $n = 36010$ input-reference pairs for testing and used the remaining dataset for training.

3. RESULTS

In the format used (Table 1) the reference sometimes had zero words, making the WER undefined, so the results below measure the performance of the model in terms of the mean Levenshtein distance (LEV).

model	LEV	Test sentences number	Correct Predicted
LAI	1.009	3601	0
MLN	1.001	3601	25
MLN finetuned	0.348	3601	2405

Table 2. Results of training compared with baseline models by mean Levenshtein distance for each correction.

As it can be seen from the table above, just finetuning on a new dataset has increased the performance of multilex norm model by 96 times if we look at the number of sentences corrected and has increased by 65 % if we look at the WER improvement.

3.1. Selected corrections

Some examples of the improvement of the model after finetuning are shown below.

Input	det fandtes ikke i barns udvalg
Expected	barens
Output (untrained)	barns
Output (trained)	barens

Table 3. An example where the trained model corrected a typo where the untrained model did not; this occurred a lot.

Input	uoverensstemmelserne i fjor er givetvis en del af forklaringen på det skarpe tonefald
Expected	uoverenstemmelserne
Output (untrained)	uoverensstemmelsern
Output (trained)	uoverenstemmelserne

Table 4. An example where the reference was incorrect but the trained model learned to match the reference, perhaps from other examples of the same word.

Input	som tre årig indledte han en livlig og aktiv pensionisttilværelse
Expected	tres
Output (untrained)	tre
Output (trained)	tres

Table 5. A seemingly advanced correction by the model.

Input	avisernes forandring påvirker ikke bare bog anmeldelsens form men også den attituder og dens retoriske greb
Expected	dens
Output (untrained)	den
Output (trained)	denne

Table 6. In this example the model was unable to learn the correct grammar through training.

4. DISCUSSION

From the decrease in error (Table 2), and examples like Table 3, it is evident that the model learned a lot by finetuning. The out-of-the-box MLN model did not perform much better than LAI, even though it had been trained on many Danish examples. It is possible that these examples were not close enough to ours, or contained different types of normalizations.

Input	hvis vi får styrket tekniker uddannelserne betyder det at de kan sætte job som i dag bliver udført af ingeniører
Expected	tage
Output (untrained)	sætte
Output (trained)	sætte

Table 7. In this example the reference text seems subjective, so it is understandable the model doesn't learn to correct.

It seems that the normalization task depends a lot on how the reference is created. In Table 7 for example, the input does not seem incorrect but the reference would prefer a different word. Even though the model does not learn the preference of the reference in this case, it generally will as it trains on the data. In Table 4 the model seems to have learned an incorrect spelling of a word from the reference. This shows that the regularization task is not well-defined and models trained on one dataset may not perform well correcting others as standards change. However, our results show that finetuning on new data is a way to learn a new regularization style.

Table 4 is an example where it may seem like the model is making an advanced correction based on the context of the text, but it could likely be using information learned from other examples in the same material. If the goal is to make corrections in the same material this is OK, but otherwise it could be seen as a case of overfitting. Given the application of this model to correct speech transcripts from a set database this would be a non-issue as the model was developed to be fine tuned for application on a given danish dataset. The finetuning operation of the ByT5 model would however have to be repeated before applying the model to a new dataset.

Examples like 6 seems like an easy grammatical correction the model should have been able to make, but it may have been confused by the grammatical mistake earlier in the sentence. This might be a disadvantage of the input style we are using, where an input is created from every word in the sentence. If a word is corrected early in the sentence, it might make sense to use the correction when correcting words later in the same sentence. A method like this would be more complicated to implement training for as it would require to update each upcoming input word in the sentence before proceeding with training.

Some of the things we could be added further to improve this model could be adding layers on top of the current model. For eg: A LSTM layer. After the addition, we could freeze the underlying model and use the new data to train the additional layers and capture positional characteristics in sentences which are particularly important in Danish as it usually follows a structure where the positioning of the subject, verb, adverb, adjective, object, time and place positioning in the main and subordinate clause is largely predictable. This could help model learn faster positional corrections.

Also currently, we are creating a new training input for each word in the audiobook sentences as demonstrated in the paper earlier. This can be further improved by using different sentinel tokens for each position in the sentence which could further help stress the significance of positional learning in Danish and also reduce the size of our training input and make the model learn faster.

Another thing that could be added is an increased use of regularization which could help our model to not overfit and would help the model in making better predictions in cases the new text would not be close enough to the model training data. This would also decrease the need to further finetune the model each time you use a new dataset.

5. CONCLUSION

We put forward our work in using transfer learning to solve the noisy speech correction task. We used a byte type model for this task and used an already pretrained model(The Danish MultiLexNorm model) to achieve good performance. This approach highlighted the importance of byte level models and transfer learning when it comes to noisy speech correction task.

Our system achieved high degree of improvement when trained on a new formalized Danish dataset compared to the original multi-lexnorm model. This demonstrates that applying transfer learning on a pretrained transformer is the key to create a computationally inexpensive language processing model. Furthermore this approach inherently requires relatively little training and data as compared to training a Deep Learning model from the bottom. Also, showing that byte type models which produce sequence of bytes perform remarkably well on noisy text data.

Another key learning was that training a pretrained model without adding any extra layers can also be enough when using transfer learning for this task.

The results achieved with the finetuned ByT5 model show the potential to produce a suitable model to generate coherent text from noisy speech transcripts.

6. REFERENCES

[1] David Samuel and Milan Straka, "UFAL at MultiLexNorm 2021: Improving multilingual lexical normalization by fine-tuning ByT5," in *Proceedings of the 7th Workshop on Noisy User-generated Text (W-NUT 2021)*, Punta Cana, Dominican Republic, 2021, Association for Computational Linguistics.

[2] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel, "Byt5: Towards a token-free future with pre-trained byte-to-byte models," 2021.