Cross-Lingual Summarization of Notice to Air Missions (NOTAMs)

Stanford CS224N Custom Project

Zixi Liu
Department of Aeronautics and Astronautics
Stanford University
liu1322@stanford.edu

Abstract

Notice to Air Missions (NOTAMs) is a notice to pilots which contains safety-critical information related to flight operations such as aeronautical hazard and surface conditions of landing runways. NOTAMs can be difficult to understand for non-trained users because it has a unique language using special contractions and domain-specific vocabularies. Our goal is to extract important information from NOTAMs in order to reduce the workload for pilots and provide the information to more potential users such as airlines and service providers. In this project, we designed a transformer based model to convert NOTAMs into a human-readable language and summarizes the information. The final design achieves 0.66 ROUGE-L score on NOTAMs related to runway and taxiway closures. In this paper, we will talk about the architectures, experiments and results of our final design. This involves training a new tokenizer for NOTAMs since it is very different from the standard English, and designing a language model to perform cross-lingual summarization from NOTAMs to English.

1 Key Information to include

- Mentor: Caleb Ziems
- External Collaborators (if you have any): None
- Sharing project: None

2 Introduction

This project is motivated by the importance of Notice to Air Missions (NOTAMs) as it contains essential information about potential hazards along the flight route that could affect the flight. NOTAMs can be difficult to understand for non-trained users because it has a unique language using special contractions and domain-specific vocabularies. Online translation tools and NOTAM commonly used contraction document can be used to decode information, however, this approach is limited by looking up words in the International Civil Aviation Organization (ICAO) dictionary and hence can contain mistakes due to word for word translation and can be costly in time. Our goal is to design a transformer based model to automatically convert NOTAMs into a human-readable language and extract important information from NOTAMs. This can help reduce the workload for pilots and provide the information to more potential users such as airlines or service providers. The final design achieves 0.66 ROUGE-L score on NOTAMs related to runway and taxiway closures which involves training a new tokenizer for NOTAMs and designing an encoder-decoder model.

3 Related Work

Several groups have investigated the use of Natural Language Processing (NLP) on decoding NOTAMs. Researchers from Airbus AI Research Team has designed a knowledge extraction pipeline which is able to classify and get information from NOTAMs (Arnold et al., 2022). This pipeline involves pretrianing unlabeled NOTAMs using RoBERTa, and finetuning on three downstream tasks including criticality prediction, named entity recognition (NER), and machine translation from NOTAMs to structured language (Airlang). Researchers from Lucerne University of Applied Sciences and Arts has trained a transformer model to convert the messages into a standardized phraseology and removes irrelevant information. The work in our paper builds on pretrained models and expands that capability by fine-tuning tokenizers for NOTAMs specific and evaluate the proper choices of hyperparamters.

4 Approach

4.1 Tokenizer

First step is to train a new tokenizer for NOTAMs because it is very different from the standard English. We want to avoid starting entirely from scratch, therefore, we evaluated and fine-tuned following types of tokenizer using pretrained models from the Tokenizers library (Wolf et al., 2019).

1. **Byte-Pair Encoding (BPE)**
   BPE (Sennrich et al., 2015) relies on a pre-tokenizer that splits the training data into words. It then uses the set of unique words and the corresponding frequency to create a base vocabulary which consists all symbols that occur in the set of unique words. BPE then learns merging rules to get a new symbol from two symbols of the base vocabulary. It does so until the vocabulary has attained the desired vocabulary size. The benefits of BPE include representing any out-of-vocabulary words using subword units that are part of the vocabulary, and handling variable-length representations of words.
2. **WordPiece**
   WordPiece(Schuster and Nakajima, 2012) is similar to BPE but rather than choosing the most frequent symbol pair, it chooses the one that maximizes the likelihood of the training data once added to the vocabulary. It computes a score for each pair using the following formula:

$$score = \frac{(freq\_of\_pair)}{(freq\_of\_first\_element * freq\_of\_second\_element)} \tag{1}$$

3. **Unigram**
   Unigram (Kudo, 2018) initializes its base vocabulary to a large number of symbols and progressively trims down each symbol to obtain a smaller vocabulary. For each symbol in the vocabulary, the algorithm computes how much the overall loss would increase if the symbol was to be removed from the vocabulary. This process is repeated until the vocabulary has reached the desired size. The loss is defined as following: Assuming that the training data consists of the words $x_1, ..., x_i, ..., x_N$ and the set of all possible tokenizations for a word x_i is defined as $S(x_i)$, then the overall loss is

$$\mathcal{L} = -\sum_{i=1}^{N} log(\sum_{x \in S(x_i)} p(x)) \tag{2}$$

4.2 Model architecture

In this project, we used the transformer encoder-decoder model (Vaswani et al., 2017) to perform summarization of NOTAMs from special domain-specific contractions to plain language. Figure 1 shows the model architecture. During the experiments, we evaluated different choices of pretrained encoder-decoder models from the Transformer library (Wolf et al., 2019) and selection of hyperparameters including numbers of hidden layers, numbers of attention heads, numbers of hidden units, attention mechanisms and activation functions.

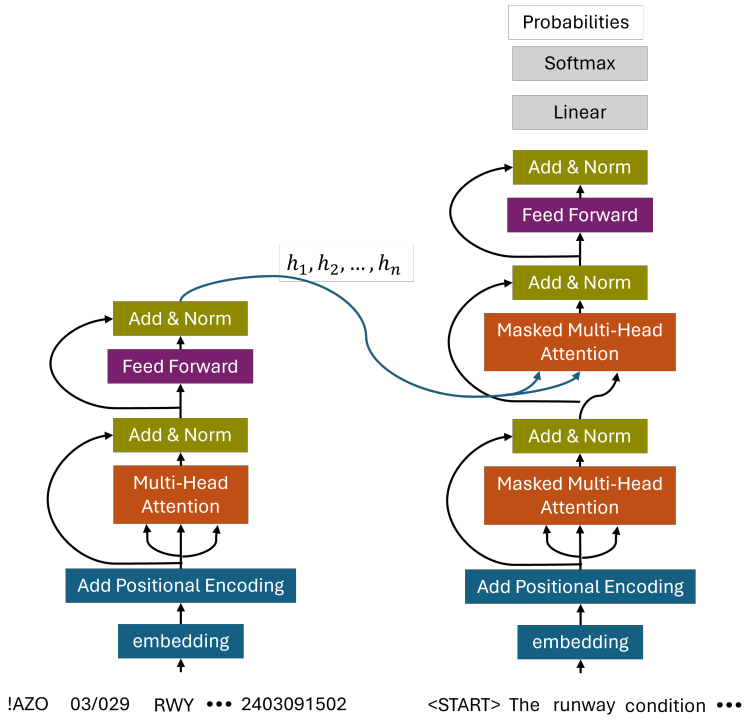

Figure 1: Transformer encoder-decoder model architecture.

5 Experiments

5.1 Data

The input dataset are NOTAMs downloaded from Federal Aviation Administration NOTAM Search Center. The output dataset are corresponding NOTAMs summaries which are manually generated by experts with verification. Figure 2 is an example of the input and the output. At current stage, we were only able to generate 2,000 sets of NOTAM-to-summary pairs. We didn't use online tools for translation because we need to make sure the NOTAMs summaries in the training dataset are 100% accurate and contains sufficient information. The best online NOTAMs translation tool is ChatGPT(OpenAI, 2024). Figure 3 shows an example translation result. in which "Bird/wildlife Aircraft Strike Hazard (BASH) Phase II", without further explanation, is helpful only to military-trained pilots who are familiar with the BASH program. By manually translating the NOTAM, we can explain that "Phase II" indicates increased bird/wildlife activity due to factors such as historic migration or nesting patterns and there is another "Phase I" which indicates reduced activity. Similar thing for "KMIB (the location identifier for a specific aerodrome)". KMIB refers to Minot Air Force Base located in Minot, North Dakota, instead of translating the 4-letters ICAO Airport Code into which airport it is, ChatGPT only states that it is an identifier for a specific aerodrome.

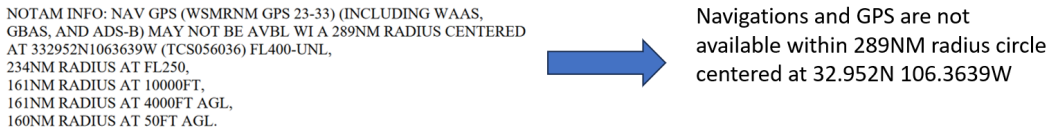

Figure 2: An example of the input and output datasets.

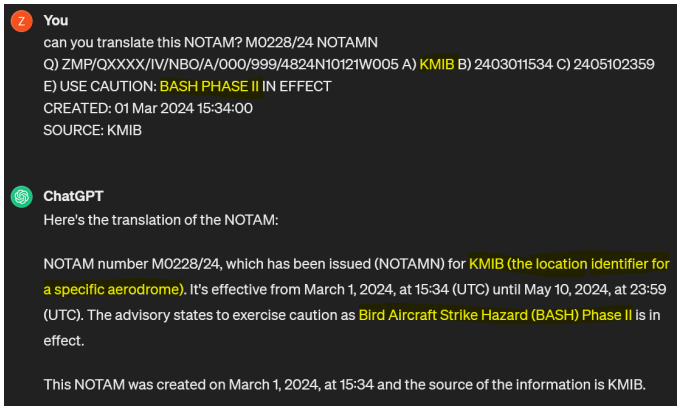

Figure 3: An example of the input and output datasets.

5.2 Evaluation method

1. **Evaluation of tokenizer**
   We evaluate tokenizers based on the numbers of matched tokenization in terms of word separation between original NOTAMs and tokenized NOTAMs. Assuming a given NOTAM, we first separate it into words using white space, then separate each word's non-alphanumeric characters from alphanumeric characters and the result is a list of desired word tokens $words_{ref}$. In terms of tokenization result, we remove white spaces and special characters used in the tokenizer, and the result is a list of tokenization outputs $words_{tokenizer}$ which can be compared with $words_{ref}$. To avoid repeatedly counting when the same word appears multiple times in a sentence, only the list of unique tokens will be selected. Then the tokenization result is evaluated using following equation:

$$score = \frac{(numbers\_of\_matched\_words)}{numbers\_of\_words\_in\_words_{ref}} \tag{3}$$

2. **Evaluation of encoder-decoder model**
   We use Recall-Oriented Understudy for Gisting Evaluation (ROUGE) to perform quantitative evaluation (Lin, 2003). There are three types of ROUGE metrics, in this project, we used ROUGE-L (Longest Common Subsequence) which is a weighted average of precision and recall. Since NOTAMs are safety-critical which requires a much more conservative metric, we will add more weights to the false positives (precision). ROUGE-L is compute as follows:

$$R_{lcs} = \frac{LCS(X,Y)}{m}, \quad P_{lcs} = \frac{LCS(X,Y)}{n}$$
$$F_{lcs} = \frac{(1+\beta^2)R_{lcs}P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{4}$$

   Where X is a reference sentence with length m and Y is a model result with length n, LCS(X,Y) is the maximum length of a common subsequence of those two sentences. F_{lcs} has values in the range of 0 to 1, where 0 means no similarity and 1 means a complete match. β is the parameter to control how much weight to put on precision.

5.3 Experimental details

The evaluation dataset is designed by first separating the NOTAMs based on main categories (domestic, flight data center, international, and military) which helps us understand if the model performs worse or best on specific type. Then we separate data into 70% pairs in the training set, 10% in the validation set, and 20% in the test set.

1. **Tokenizer**
   We compared fine-tuning results from different pretrained tokenizer models from the Tokenizers library (Wolf et al., 2019) including Byte-Pair Encoding (BPE): e.g. GPT-2,

RoBERTa, WordPiece: e.g. BERT, SentenMariancePiece in combination with Unigram: e.g. ALBERT and T5. All tokenizers are trained on both NOTAMs and the summaries in English for encoder and decoder model separately. The score is calculated using equation 3. We evaluate tokenizer based on worst, average, and best performance.

2. **Encoder-decoder model**
   All parameters of the encoder and decoder were chosen based on test ROUGE-L score using equation 4. This project tested numbers of hidden layers, numbers of attention heads, numbers of hidden units, attention mechanisms and activation functions. For instance, among numbers of attention heads 10, 12, 14 and 16, result showed that 12 attention heads led to the lowest evaluation loss and highest ROUGE-L score. Same comparison method was used on the other parameters which led to our final model: for both encoder and decoder, it uses GELU (Gaussian Error Linear Unit) as hidden activation function with 10% dropout probability, 1536 hidden units, 14 hidden layers and it uses multi-head self-attention mechanism with 12 attention heads.

5.4 Results

1. **Tokenizer**
   Figure 4 shows a comparison between different tokenizers. Notice that BPEs tend to have better performance than other types of tokenizers, and GPT-2 performs slightly better. This is as expected because BPE's pre-tokenization is already based on white space, and during training, it learns merging rules which forms new symbol from two symbols of the base vocabulary based on the appear frequency. This turns out to match quite well with the way that special contractions are formed in NOTAMs. Therefore, we used GPT-2 tokenizer in our encoder-decoder model which improves the overall model performance as well.

Category	Byte-Pair Encoding (BPE)				WordPiece		SentenMariancePiece			
Name	GPT-2		RoBERTa		BERT		ALBERT		T5	
Corpus	*NOTAM*	*Translation*	*NOTAM*	*Translation*	*NOTAM*	*Translation*	*NOTAM*	*Translation*	*NOTAM*	*Translation*
Worst Result	0.69	0.75	0.63	0.75	0.21	0.5	0.14	0.29	0.34	0.42
Average Result	**0.80**	**0.96**	0.78	0.96	0.44	0.79	0.28	0.54	0.53	0.65
Best Result	0.95	1	0.95	1	0.63	0.95	0.47	0.77	0.85	0.87

Figure 4: Comparison of accuracy scores for different tokenizers.

2. **Encoder-decoder model**
   Figure 5 shows a comparison between different values of hyper-parameters. Noticed that the ROUGE-L scores are relatively low, this is due to adding more complicated training datasets and training the model on limited numbers of NOTAM-to-summary pairs (2,000 sets). The model cannot successfully learn those cases, the details will be discussed in the following section. The baseline model uses the default values from the Transformers library and our final choices slightly improves the overall model performance due to training and evaluating on NOTAMs corpus.

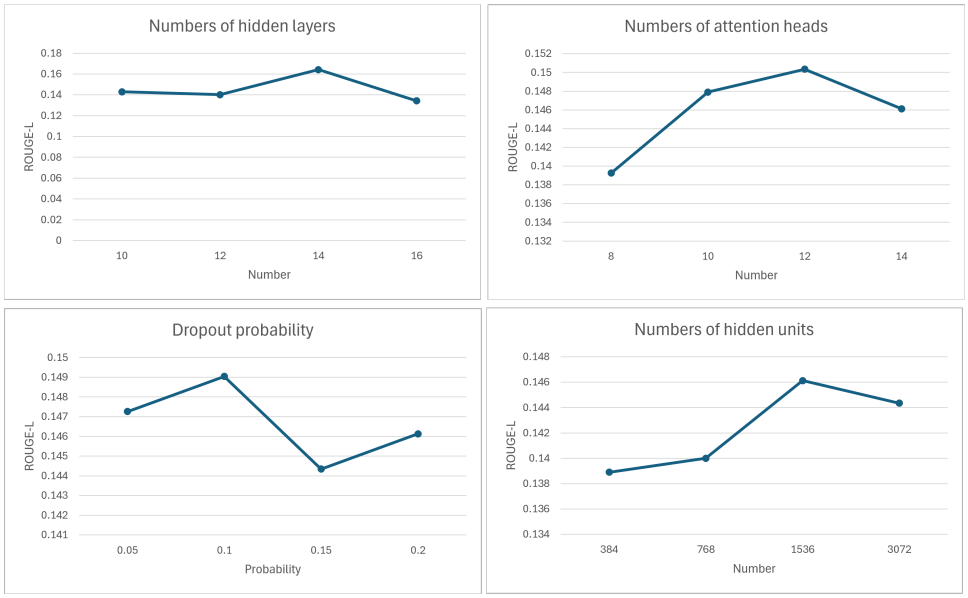

Figure 5: Comparison of ROUGE-L scores for different values of hyperparameters.

6 Analysis

1. **Tokenizer**
   Figure 6 shows the improvement we have seen by training a new tokenizer. One example is: the word "TWY" in NOTAMs refers to "taxiway", the original tokenizer splits that into 'TW' and 'Y' but the new tokenizer learns to treat it as one word 'TWY'.

- **Example NOTAM:**

!DEN 10/112 DEN TWY EE BTN **TWY** M AND TWY ED **4795**FT X 75FT CONC **LGTD COMMISSIONED** 2310051653-PERM

- **Old default tokenizer: (GPT2)**

['!', 'DEN', 'Ġ10', '/', '112', 'ĠDEN', 'ĠTW', 'Y', 'ĠEE', 'ĠBT', 'N', 'ĠTW', 'Y', 'ĠM', 'ĠAND', 'ĠTW', 'Y', 'ĠED', 'G4', '795', 'FT', 'GX', 'G75', 'FT', 'GCONC', 'GLG', 'TD', 'GCOMM', 'ISSION', 'ED', 'G23', '100', '516', '53', '-', 'PER', 'M'] (*with length 37*)

- **Retrained tokenizer:**

['!', 'DEN', 'Ġ10', '/', '112', 'ĠDEN', 'ĠTWY', 'ĠEE', 'ĠBTN', 'ĠTWY', 'ĠM', 'ĠAND', 'ĠTWY', 'ĠED', 'Ġ4795', 'FT', 'GX', 'G75', 'FT', 'GCONC', 'GLGTD', 'GCOMMISSIONED', 'G2310051653', '-', 'PERM'] (*with length 25*)

Figure 6: Tokenization results from original GPT-2 tokenizer and retrained tokenizer on NOTAMs corpus.

2. **Encoder-decoder model**

   Figure 7 shows an example case when the model succeeds. We realized that our model can succeed mostly on NOTAMs with short sentence and usually relates to information about runway and taxiway closure. One potential reason could be the imbalanced NOTAMs types in our training dataset. In addition, this example output only achieves 0.53 ROUGE-L score even though the overall result is readable and matches the meaning. This indicates another reason for seeing low ROUGE-L score in the experiment results, which is the difficulty for the model to predict additional verbs or nouns to make the translation grammatically correct.

• **Example NOTAM:**
!DCA 03/288 DCA RWY 04/22 CLSD
• **Human translation:**
Runway 13/91 at Washington National Airport (DCA) is closed
• **Model:**
DCA runway 13/91 closed

Figure 7: An example of successful summarization from the trained model.

Our model was trained on a limited dataset which leads to some performance issues. We noticed that there are some common cases when the model fails. The first common case is when the reference output contains extra words from experts who decided to polish the translation output. Examples include adding prefix of "This NOTAM is updating/stating/alerting" before providing the actual information from the NOTAMs context. The second case is when we want to force the model to decode information from special terminologies such as "Phase II" to "increased bird/wildlife activity due to historic migration or nesting patterns", and "KDEN" to "Denver International Airport". This is no longer a word to word translation, it requires tons of training datasets for the model to learn the underlying relations and meanings. The third common case is when the NOTAM contains location coordinates such as "41°49'41" N". We believed that these pairing results can be achieved only if we have sufficient numbers of training dataset such as millions of NOTAM-to-summary pairs. However, at current stage, forcing model to train on detailed translation outputs can only lead to incorrect results.

7 Conclusion

In this work, we presented the use of transformer encoder-decoder model to extract knowledge from NOTAM aeronautical messages. We showed that training a new tokenizer for NOTAMs is necessary and can improve the model performance. In addition, a pretrained model can be efficiently reused on downstream tasks with dedicated fine-tuning. Our model can work reasonably well on summarizing NOTAMs related to runway and taxiway closures with an average of 0.66 ROUGE-L score. However, our current training result does not perform well on other types of NOTAMs due to the lack of training dataset (NOTAM-to-summary pairs). One potential solution to this problem, besides generating more NOTAM-to-summary pairs, is to divide the problem into sub-tasks and train different models for different tasks. For instance, decoding all 4-digits ICAO airport code into full airport name can be a difficult sub-task. In the future, in addition to tunning hyperparameters of the encoder-decoder model, different selections of pretrained models can also be evaluated.

References

Alexandre Arnold, Fares Ernez, Catherine Kobus, and Marion-Cécile Martin. 2022. Knowledge extraction from aeronautical messages (notams) with self-supervised language models for aircraft pilots. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track*, pages 188–196.

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

Chin-Yew Lin. 2003. Rouge: Recall-oriented understudy for gisting evaluation.

OpenAI. 2024. ChatGPT. [Computer software]. Accessed: 17 March 2024.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE.

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

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. arxiv. *arXiv preprint arXiv:1910.03771*.