Enhancing Named Entity Recognition with Data Augmentation via Abstractive Summarization

Douglas Bowen
Dept. of Mathematics
Wilfrid Laurier University
Waterloo, Canada
bowe3920@mylaurier.ca

Yanan Chen
Dept. of Phys. and Comp. Sci.
Wilfrid Laurier University
Waterloo, Canada
chen0040@mylaurier.ca

Dr. Yang Liu
Dept. of Phys. and Comp. Sci.
Wilfrid Laurier University
Waterloo, Canada
yangliu@wlu.ca

Dr. X. Sunny Wang*
Dept. of Mathematics
Wilfrid Laurier University
Waterloo, Canada
xwang@wlu.ca

***Abstract*—Named Entity Recognition is a natural language processing task that requires token-level data for training. In some domains, little to no training data currently exists and as such, individuals must go through the time-consuming and costly process of creating training data. Though some common tag sets and domains have plenty of data compiled by the public (e.g. Wikipedia, Twitter, News), custom named entity tags and niche domains often have little to no data available, making training quite challenging. As a result, steps must be taken to create a large dataset from scratch or a small low-resource set. Data augmentation is a technique that attempts to alleviate this. Through various approaches, individuals can create artificial data to supplement the original and help increase model performance. This paper examines various sentence-level approaches to data augmentation in low-resource domains wherein artificially produced training data is required for decent model training. We simulate a low-resource scenario for data augmentation, and a new generation approach using abstractive summarization is proposed to create sentences with new structures from the original data. In our experiments, we mimic four low-resource scenarios for examining model performance with abstractive summarization as compared to other augmentation methods and the original un-augmented data. Results show that abstractive summarization provides a boost to model results as compared to the original un-augmented data.**

***Index Terms*—data augmentation, named entity recognition, abstractive summarization, low-resource, pre-trained language model**

I. Introduction

Natural Language Processing (NLP) and other deep learning tasks tend to require a large amount of training data. In the case of natural language processing, pre-trained models can be custom tailored to different tasks. However, these models still require a sizeable amount of training data to yield good results. Furthermore, the more customized the task, the more data that is often required. In some niche domains, there may be no existing training data and thus would require individuals with domain-expertise to perform the time-consuming task of creating training data.

Data augmentation techniques for sentence-level NLP tasks (such as classification models) augment individual sentences in some way to artificially inflate the amount of training data that can be used. This inclusion of additional augmented data helps to increase model performance without requiring experts with domain knowledge to spend time preparing more training data. In the case of named entity recognition (NER) models, tokens within each sentence are predicted to be an entity class or not. As a result, consideration must be given in how to apply augmentation techniques to a sentence without unjustly altering token-entity tag pairings.

Current researched methods include augmentation that is applied both via logical rule-based methods [1] and more involved methods that range from using a simple Long Short Term Memory (LSTM) neural network [2] to more involved deep-learning options such as paraphrasing [3] and back-translation [4] methods.

Though these current techniques for data augmentation are widely applicable, in cases where sentences come from a larger paragraph or article, valuable information is obscured when using sentence-level augmentation methods. In this paper, we attempt to provide a new augmentation method that can be applied to multiple sentences at once as opposed to singular sentences. Abstractive summarization is one such technique that can be applied to the paragraph-level, resulting in a summary constructed of novel sentences wherein the structure of the generated summary is unique from the sentences in the original paragraph. We empirically compare various methods including previously researched and proven techniques to serve as a baseline for validation. We demonstrate that the proposed paragraph-level augmentation method of abstractive summarization performs effectively on a manually annotated corpus constructed of Wikipedia articles that have had named entities tagged. This research provides empirical validation that abstractive summarization can be a new data augmentation method to be utilized in the case of paragraph-level data. This method provides yet another tool for generating artificially created sentence-level data, but can be applied at the paragraph-level and is still applicable to models that require token-level information, such as NER models. Further, this method can be used in tandem with other augmentation techniques as sentence structure is unique.

The remainder of the paper is structured as follows. Section II presents the related works that are focused on augmenting data for named entity recognition. Section III discusses the approach taken to apply abstractive summarization on paragraph-level data and produce augmented data at the sentence-level.

Funding provided by: NSERC, Mitacs, SOSCIP, The Quantum Insider

Section IV reviews the dataset utilized, implementation of process pipeline, and discusses the results of our research. Finally, Section V concludes our paper, while Section VI provides acknowledgment to those who also contributed to our work.

II. Related Work

In this section, we focus on examining data augmentation techniques related to NER, a subset of more broad NLP tasks. Previous research into augmenting data with named entities can be categorized in two ways:

1) **Rules-Based Approach:** The simplest approach is one that focused on tweaking traditional rules-based NLP approaches to DA so that they may be applied to NER situations [1]. Four underlying rules-based methods for augmenting NER data were constructed, for which combinations of these methods could also be utilized. Each of the following methods utilizes a probability distribution to determine whether or not the suggested transformation should take place on the specified token/segment. A success rate p is utilized in the Binomial Distribution (specifically, the Bernoulli Distribution for n=1 trial).
    a) The first method discussed is Label-Wise Token Replacement (LWTR) [1] wherein each token within a sentence is evaluated randomly for success or failure. If the randomly generated number outputs a success, the token is replaced with a randomly selected token designated with the same label. Otherwise, no replacement occurs and the next token is evaluated in the same way.
    b) The second method discussed is Synonym Replacement (SR) [1]. In a very similar method as to LWTR, each token within a sentence is evaluated randomly. In the case of SR however, if the randomly generated result indicates a success, the token is replaced with a synonym of itself. For synonyms that are multi-word, the associated label is applied in the standard B-Entity, I-Entity order.
    c) The third method discussed is Mention Replacement (MR) [1]. If a token is marked for replacement, an entry with the same entity type (but of any length) is swapped into its place. The new token will then utilize its own labels, even if longer than the original token.
    d) The final method discussed is substantially different from LWTR, SR, and MR [1]. Shuffle within Segment (SIS) evaluates each continuous segment of labels within a sentence randomly for success or not. In the case of success, the segment will be randomly shuffled to alter the order of tokens, but not labels.

    Lastly, it is possible to use any combination/mixture of the above methods for a more comprehensive approach to data augmentation.
2) **Generative Model Based Approach:** Another approach involves utilizing neural networks to augment sentences. There are three techniques examined.
    a) The first method utilizes a simple Long Short Term Memory (LSTM) model which is trained on the sentence-level that has been linearized to have entity tags in front of the tokens. Using this model, augmented sentences are generated and then delinearized [2].
    b) The second method performs a back-translation on the sentence-level from English to German and back [4]. The sentence is first split into segments of continuous tags so that each segment corresponds either to context or an entity. Segments that correspond to context are then translated to German and back only if they consist of length three or more. Entity segments are not translated. Then the segments are re-combined to create an augmented sentence.
    c) The last method runs sentences through a paraphrasing model [3]. This model would then generate the requested number of paraphrased sentences as augmented data to for NER models to train on.

III. Method

While the aforementioned literature focuses on various methods of sentence-level augmentation, this type of augmentation does not significantly alter the linguistic structure of sentences. Furthermore, there is a limitation to how many variations can be made when utilizing some of these methods (i.e. the synonym replacement method only has so many potential tokens that can be replaced with synonyms). As such, we examine a method of augmentation focusing on the paragraph-level to create unique augmented sentences that differ in linguistic structure from the original and that can potentially be used in tandem with other methods.

We propose that a simple, pre-trained abstractive summarization model can provide augmented sentences that subsequently increase NER model performance when utilizing these additional generated summaries as augmented sentences. Abstractive summarization is a method in which a model generates a summary with linguistically unique, novel sentences. These sentences can overlap with those from the source text, but often have their own additions combining information from multiple sentences into one. While extractive summarization also exists, this method of summarization simply concatenates sentences from the source text without any changes. Sentences in the source text are ranked by relevance and the summary is constructed of the most relevant sentences, copied word for word one after another. An example comparing these methods can be seen in Table I. As such, extractive summarization provides no benefit in data augmentation and was not examined Section III-A will discuss the differences between potential pre-trained summarization models and the process for selecting which model to be used, while Section III-B discusses common issues that occur with generated summaries.

TABLE I
COMPARISON OF EXTRACTIVE AND ABSTRACTIVE SUMMARIZATION

Sample Source Text	
010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets . This album proved to be more commercial and more techno-based than Osc-Dis , with heavily synthesized songs like Introduction 010 and Come . Founding member Kojima Minoru played guitar on Good Day , and Wardanceis cover of a song by UK post punk industrial band Killing Joke . XXX can of This had a different meaning , and most people did n't understand what the song was about . it was later explained that the song was about Cannabis (' can of this ' sounding like Cannabis when said faster) it is uncertain if they were told to change the lyric like they did on P.O.P and HUMANITY . UK Edition came with the OSC-DIS video , and most of the tracks were re-engineered .	
Extractive	**Abstractive**
010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets .	010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets with heavily synthesized songs like Introduction 010 and Come, and covers of a song by UK post punk industrial band Killing Joke

Section III-C reviews the entire pipeline required to take a paragraph or article and select an appropriate number of augmented sentences.

A. Summarization Model Selection

Many publicly pre-trained transformer models can perform summarization, of which most are hosted on HuggingFace's Transformers Library [5]. Three common models are examined, these being:

1) BART [6]
2) T5 [7]
3) PEGASUS [8]

Of the above models, BART and T5 are Seq-2-Seq structured transformers, while PEGASUS utilizes an underlying T5 model with training on data specifically for abstractive summarization. BART and T5 can be utilized for text generation, translation, Q&A, and summarization, among other things, while PEGASUS is geared more towards only abstractive summarization.

The BART transformer structure was created by Facebook. The structure is an expansion on BERT, which at its core is a bi-directional encoder which was trained via masked language modelling. BART furthers this by adding in an autoregressive decoder to provide further functionality over BERT and access to standard encoder-decoder tasks. Training of the model first corrupted source text via noising function and then learned to reconstruct the corrupted text as close to source-level as possible.

Similar to BART, the T5 transformer structure was created by Google and released within the same week. It was also an expansion from BERT and thus has the same bi-directional encoder as BART, and further adds its own auto-regressive decoder. The primary differences in T5 and BART are the structure of layers within the decoder. Another slight difference is that the training method used by T5 is instead a mix of variations used to train for different tasks rather than the standard corruption/fill-in-the-blank training approach.

Lastly, PEGASUS was created by Google and based off the original T5 Seq-2-Seq transformer structure. Its goal was to fix other models' weaknesses in summarization generation, wherein results were often extractive and not abstractive. PEGASUS pre-trains the T5 model on a large text corpus wherein important sentences get masked from the input source and then generated utilizing the remaining sentences. As a result, this pre-trained model is quite strong at abstractive summarization specifically, but cannot be utilized as well for other Seq-2-Seq tasks.

To select an appropriate summarization model, we randomly select twenty-five articles taken from the dataset described in Section IV-A and evaluate the performance along four dimensions that can potentially effect the performance of generation. The generated summary for each of these was examined for outputs that were:

1) Extractive: generated summary was constructed with sentences pulled directly from the source text, without alteration.
2) Repetitive: generated summary displayed some degree of repetition wherein sequences of tokens near the end of the summary were repeated twice or more.
3) Hallucinated: generated summary contained entities that were not present in the source text.
4) Incoherent: generated summary was incoherent in that the sentence(s) lacked any semblance of structure or flow.

The experimental result is shown in Table II. While the above output types are all sub-optimal, repetitive outputs are preferred over hallucinated ones, and incoherent and extractive outputs were considered to be most impractical. If the output did not fall into one of these categories, it was considered satisfactory. The three summarization models were further examined with a variety of hyper-parameters changed in the hope of making a more comprehensive analysis. From Table II, it is clear that the PEGASUS model outperforms the others in terms of abstractive summarization. Furthermore, setting the minimum length to the number of tokens in the source text and the number of beams to thirty-two provided similar results to not having a number of beams set, but provided more variety in summaries. As a result, the PEGASUS model was selected with hyper-parameters setting the minimum length to the number of tokens in the source text and beams to thirty-two.

B. Observations on Text Summarization

When generating summaries, two common issues tend to occur that can hinder how appropriate a summary is. Previously mentioned in Section III-A, these are cases of entity hallucination and repetition. While incoherent and extractive outputs are also an issue, the selected model did not have this issue when sampling twenty-five articles (as seen in Table

TABLE II
GENERATED SUMMARY SUCCESS RATES

Parameters	Model	Satisfactory	Extractive	Repetitive	Incoherent	Hallucination
Default	BART	0/25	19/25	0/25	6/25	0/25
	T5	1/25	19/25	0/25	5/25	0/25
	PEGASUS	17/25	7/25	0/25	1/25	0/25
MinLength=Article	BART	0/25	22/25	0/25	3/25	0/25
	T5	1/25	22/25	0/25	2/25	0/25
	PEGASUS	18/25	0/25	5/25	0/25	2/25
MinLength=Article Top P=0.9	BART	0/25	22/25	0/25	3/25	0/25
	T5	0/25	24/25	0/25	1/25	0/25
	PEGASUS	11/25	0/25	6/25	0/25	8/25
MinLength=Article Num Beams=32	BART	0/25	22/25	0/25	3/25	0/25
	T5	0/25	23/25	0/25	2/25	0/25
	PEGASUS	17/25	0/25	6/25	0/25	2/25
MinLength=Article Num Beams=32 RepetitionPenalty=2.0	BART	0/25	22/25	0/25	3/25	0/25
	T5	0/25	23/25	0/25	2/25	0/25
	PEGASUS	17/25	0/25	4/25	0/25	4/25

II) and as such is not a concern. When considering that these summaries are being utilized to augment training data for a NER model, entity hallucination and repetition can be detrimental to model performance.

Motivated by such observations, we decide to follow a paradigm of "generation and filtering" to mitigate this issue. More specifically, we apply ROUGE F1 scores to generated summaries, wherein the highest scores are selected to improve the generation quality. Note that higher ROUGE scores do not inherently indicate a satisfactory summary output, but tend to be correlated and as such provide a better option for selection.

C. Proposed Method

While the first step in the process is in deciding on a pre-trained model to use (PEGASUS), generating a sufficient number of augmented sentences from summaries requires a few adjustments to be made to the source texts. In generating summaries, the order of sentences within the source text tends to determine the topic that the generated summary focuses on. For example, if an article had thirty sentences, moving the last four sentences to now be the first four could completely change the focus of the generated summary. A smaller example can be seen in Table III. Generating a summary tends to produce usually one long sentence and more rarely two sentences at most. As such, without shuffling an article with thirty sentences we would only have one or two augmented sentences to work with. However, when we apply shuffling we can create more augmented sentences, with the only limit being the number of permutations possible based on how many sentences the article contains. While an article of only two sentences has just one other ordering, an article with 5 sentences has 120 potential orderings, and similarly up to 120 different augmented sentences that can be utilized. With this in mind, any source text can have at a minimum a 1:1 ratio of original sentences to augmented sentences, if not much higher as the number of permutations increases. For this research, samples were created for each article based on how many sentences were in the source text multiplied by 3. The required samples would then be selected based on the highest ROUGE-1 F1 scores.

Once a sufficient number of generated summaries exist and the samples for augmenting are selected, the second step is breaking the generated text into tokens so that commas, brackets, and other non-alphanumeric characters could be assigned tags. This was done using the NLTK library [9] which accurately breaks texts by both sentence and token.

The final step is mapping tokens to their respective tags for the generated texts. The main approach to this involved creating a "sliding *n*-gram window" algorithm (Algorithm 1, Table IV) that first mapped the source text for token-tag pairings at each *n*-gram level up to the largest continuous entity segment. As entities were almost always capitalized, 2-gram and higher mappings were quite accurate. In the case of 1-gram mappings, a blacklist was created for common words that could also appear as entities, such as "The" and "of". To then apply the mapping, every token in the generated summary was set to a non-entity "O" tag. Next, the tokens were searched for 1-gram matches, 2-gram matches, and so on with "O" tags being replaced with the appropriate tag as matches are found. Though this technique sometimes missed entities when the words were re-ordered, this was quite rare. By iterating in an n-gram approach, cases where the generated summary shortened an entity could still be captured. Entity hallucination could also not be accounted for. In trying to accurately map entities from the source text to generated text, two types of variations on the source text were proposed in an attempt to alleviate any issues in mapping via sliding *n*-gram window. These variants were linearized and tag-replaced variations on the source text, which can be seen in Table V. However, these variations brought with them new problems. In an examination of 25 articles similar to when PEGASUS hyper-parameters were examined, the rate of incoherence and entity hallucination increased drastically for all methods as seen in Table VI. As a result, the Linearized and Full-Tag variants were not utilized, while the One-Tag and Unique-Tag variants were kept to compare against but were not expected to perform well.

The entire process can be visualized in Fig. 1.

TABLE III
SHUFFLED SOURCE TEXT SUMMARIES

Source Text	Generated Output
010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets . This album proved to be more commercial and more techno-based than Osc-Dis , with heavily synthesized songs like Introduction 010 and Come . Founding member Kojima Minoru played guitar on Good Day , and Wardanceis cover of a song by UK post punk industrial band Killing Joke . XXX can of This had a different meaning , and most people did n't understand what the song was about . it was later explained that the song was about Cannabis (' can of this ' sounding like Cannabis when said faster) it is uncertain if they were told to change the lyric like they did on P.O.P and HUMANITY . UK Edition came with the OSC-DIS video , and most of the tracks were re-engineered .	010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets, with heavily synthesized songs like Introduction 010 and Come, and covers of a song by UK post punk industrial band Killing Joke
This album proved to be more commercial and more techno-based than Osc-Dis , with heavily synthesized songs like Introduction 010 and Come . it was later explained that the song was about Cannabis (' can of this ' sounding like Cannabis when said faster) it is uncertain if they were told to change the lyric like they did on P.O.P and HUMANITY . 010 is the tenth album from Japanese Punk Techno band The Mad Capsule Markets . Founding member Kojima Minoru played guitar on Good Day , and Wardanceis cover of a song by UK post punk industrial band Killing Joke . XXX can of This had a different meaning , and most people did n't understand what the song was about	010 is the second album from Japanese Punk Techno band The Mad Capsule Markets, and the follow-up to Osc-Dis, which was the band's first album to sell more than one million copies in the UK, and was the band's best-selling album to date.

Algorithm 1 Sliding N-Gram Mapping (Psuedo Code)

```
Inputs:
    Max_NGrams = Longest Tag Segment
    Blacklist = Common 1-Gram Non-Entities
    Mapping = {All N-Gram Entities : Tag}
for each generated Summary_Token_List do
    Initialize:
    Set All Summary Entity Tags = "O"
    for i in range(1, Max_NGrams) do
        for j in range(len(Summary_Token_List) - i + 1) do
            Sequence = Summary_Token_List[j: j + i]
            if Token Sequence Exists in Mapping Dict then
                Replace Sequence Tags with Mapping Tags
            else Do Not Adjust Entity Tags
            end if
        end for
    end for
end for
```

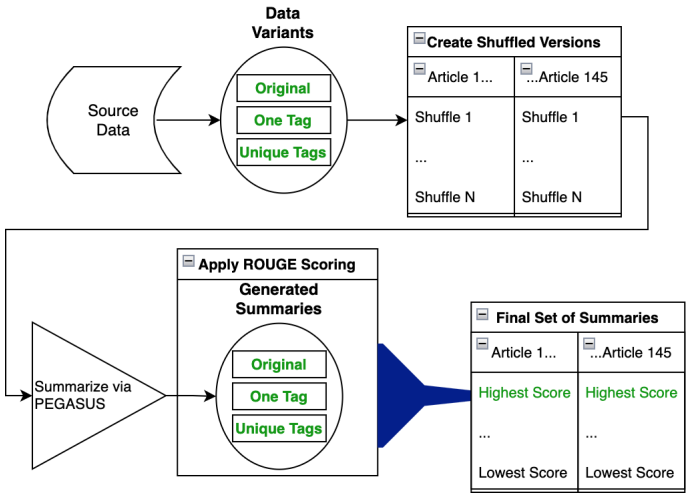

Fig. 1. Visualized Process Pipeline

TABLE IV
EXAMPLE OF SLIDING N-GRAM MAPPING

Iteration	Output
Source Text	010 is the tenth album from Japanese punk techno band The Mad Capsule Markets .
Reference Tags	MISC O O O O O MISC O O O ORG ORG ORG ORG O
Blacklist*	"The"
Mapping (1-Gram)	"010":MISC, "Japanese":MISC, "The":ORG, "Mad":ORG, "Capsule":ORG, "Markets":ORG
Mapping (2-Gram)	"The Mad":ORG, "Mad Capsule":ORG, "Capsule Markets":ORG
Mapping (3-Gram)	"The Mad Capsule":ORG, "Mad Capsule Markets":ORG
Mapping (4-Gram)	"The Mad Capsule Markets":ORG
Generated Summary	The Japanese band The Mad Capsule Markets released their tenth album 010 .
Initialization	O O O O O O O O O O O O O O O
1-Gram Iteration	O MISC O O ORG ORG ORG O O O O O O MISC O
2-Gram Iteration	O MISC O ORG ORG ORG ORG O O O O O O MISC O

*The blacklist contains words that may appear as parts of entities but are commonly found on their own as non-entities. This is indicated in red in the above table. For example, "The" is commonly found at the start of sentences. These are removed from the 1-gram mapping list and later re-mapped in the 2-gram+ mappings.

IV. EXPERIMENTS

In this section, we evaluate the success of abstractive summarization as a means for data augmentation on the WikiGold dataset [10]. Results are compared against previously mentioned rule-based and paraphrasing methods of data augmentation which serve as a baseline for comparison.

A. Dataset

The WikiGold dataset was selected for use as it met the requirements of having (1) entity tags at the token-level and (2) related sentences following one-another that form cohesive paragraph. To create an appropriate training-test split that mimicked a low-resource scenario, the dataset was split into

TABLE V
SOURCE TEXT ENTITY REPLACEMENT TECHNIQUES

Variation	Output
Original	Oleg Gazmanov is a Russian singer .
Reference Tags	PER PER O O MISC O O
Linearization	PER Oleg PER Gazmanov is a MISC Russian singer .
Full Tag Replacement	PER PER is a MISC singer .
One Tag Replacement	PER is a MISC singer .
Unique Tag Replacement	PER1 is a MISC1 singer .

batches of articles that provided 50, 100, 250, and 500 sentences, denoted "X-Small", "Small", "Medium", and "Large" batches respectively. To construct these batches, the number of sentences within each article was recorded. Next, articles with less than 2 sentences or more than 40 sentences were removed from the dataset to prevent one article entirely filling a batch. Random articles were then selected until the number of sentences was within 10% of the batch size. These articles make up the training set. The articles not selected were left as a testing set, and this process was then replicated 10 times to ensure results later on were not influenced purely by random selection. Characteristics of the batches can be seen in Table VII.

B. Experimental Settings

Articles from the replicated batches were selected to be summarized where three augmented sentences were created per original sentence in the article by selecting the appropriate number of generated samples. In the cases of the rule-based and paraphrasing methods, these were utilized to serve as a baseline for comparison with each sentence in the article having LWTR and Paraphrasing methods applied three times for a total of one original sentence and three augmented ones.

The Cased BERT model [11] was chosen to perform the task of NER. The model was run with 3 epochs, a learning rate of 2×10^{-5}, and a weight decay rate of 0.01. No hyper-parameter tuning was performed as results on 1,300 randomly selected sentences from the WikiGold dataset resulted in F1 scores over 80%. Each of the ten replications for each batch size was fed into the NER model, with results for precision, recall, and F1 being recorded. The results for the ten replications were then recorded with a 95% confidence interval.

C. Results

The results can be seen in Table VIII, wherein the highest performing F1-score is bolded. The rules-based approach results are based off of the highest performing method from LWTR, SR, and SIS for Binomial probability rates 0.1, 0.3, 0.5, and 0.7 as seen in the footnote for Table VIII.

We find that all methods for augmentation provide a boost to NER model results over the original un-augmented data. Of the augmentation methods, the baseline methods both tend to outperform summarization when applied to the original source text. In the case of paraphrasing, though it performs slightly worse on average in the X-Small batch size, it does so with a smaller confidence interval. Subsequently, it performs better at all other levels. On the other hand, rules-based augmentation (LWTR with a 10% binomial rate) performed best, outperforming all methods. Lastly, the One Tag Replacement and Unique Tag Replacement variations on the source text provided a boost to performance at low batch sizes, but was significantly outclassed by other methods, likely due to entity hallucination issues previously discussed. In performing a paired t-test to determine if abstractive summarization results differed significantly from other methods at the 5% significance level, it was found that only the paraphrased method at the small batch size could be considered equivalent with a p-value of $\sim$0.82.

D. Limitations

For the application of abstractive summarization as a means for augmenting training data, the source text is required to be constructed of multiple related sentences that form a paragraph. Furthermore, entity tags are required for every token in the article. It should be noted that this approach could be applied to sentences randomly combined to form a paragraph, but that results from doing so were not examined.

V. CONCLUSION

In this paper, we adapt an abstractive summarization model to a source text of related sentences. We show that this technique can provide a significant boost to NER model performance when utilized in low-resource scenarios, providing diverse, high quality, and unique sentence structures.

The generated summaries provide sentences that are linguistically unique when compared to both the original source sentences and augmented sentences generated from rule-based and paraphrased methods. Furthermore, information from the entire source text can be utilized as opposed to sentence-level information that other methods use.

In time, a more rigorous approach to hyper-parameter tuning should be taken while comparing larger training cycles and augment multiples. These generated summaries provide a unique opportunity in that other augmentation techniques can then further be applied to the summaries to potentially provide a larger boost.

VI. ACKNOWLEDGMENTS

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) Individual Discovery Grants (RGPIN-2018-06641 and RGPIN-2019-05917), Mitacs (IT28067), SOSCIP, and The Quantum Insider (TQI). The authors would like to thank Ian Fraser, Herteg Kohar, and Jason Lechter for their support during this project.

REFERENCES

[1] X. Dai and H. Adel, "An analysis of simple data augmentation for named entity recognition," in *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, D. Scott, N. Bel, and C. Zong, Eds. International Committee on Computational Linguistics, 2020, pp. 3861–3867. [Online]. Available: https://doi.org/10.18653/v1/2020.coling-main.343

TABLE VI
SUMMARY OUTPUT SUCCESS RATES

Model	Satisfactory	Extractive	Repetitive	Incoherent	Hallucination
Original	17/25	0/25	6/25	0/25	2/25
Linearized	4/25	0/25	3/25	8/25	10/25
Full Tag	4/25	0/25	9/25	0/25	12/25
One Tag	11/25	0/25	3/25	0/25	11/25
Uni Tag	11/25	0/25	8/25	0/25	6/25

TABLE VII
METRICS BY TRAINING TEST SPLIT

Batch Size	**X-Small**		**Small**		**Medium**		**Large**	
Set*	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**	**Train**	**Test**
# of Articles	5	140	9	136	23	122	45	100
# of Sentences	49.6	1,718.4	100.6	1,667.4	246.3	1,521.7	505.8	1,262.2
# of Non-Entities	841.2	31,734.8	1,753.1	30,822.9	4,380.9	28,195.1	9,321.7	23,254.3
# of Entities	181.1	6,249.9	406.6	6,024.4	972.2	5,458.8	2,018.7	4,412.3

*Averaged over 10 Randomized Replications

TABLE VIII
MODEL RESULTS FOR TESTED METHODS (α = 0.05, N=10)

Batch Size	Method	Precision	Recall	F1-Score
X-Small (S=50)	Original	3.33%±6.53%	0.00%±0.00%	0.01%±0.01%
	Rules*	**26.28%± 7.18%**	**16.48%± 4.79%**	**20.01%± 5.16%**
	Paraphrased	22.38%± 6.27%	8.37%± 3.08	11.68%± 3.56%
	Article	23.46%± 5.98%	8.69%± 3.65%	12.05%± 4.41%
	One Tag	15.97%± 9.63%	0.91%± 0.88%	1.64%± 1.56%
	Unique Tag	16.38%± 9.10%	1.83%± 1.47%	3.06%± 2.33%
Small (S=100)	Original	8.45%±5.56%	0.59%±0.77%	0.99%±1.23%
	Rules*	**53.77%± 3.1%**	**59.14%± 7.34%**	**56.18%± 4.84%**
	Paraphrased	47.27%± 2.58%	47.64%± 5.06	47.20%± 3.75%
	Article	43.49%± 3.83%	40.2%± 5.62%	41.61%± 4.71%
	One Tag	23.18%± 3.07%	12.56%± 3.37%	16.03%± 3.47%
	Unique Tag	25.32%± 2.35%	15.52%± 3.33%	18.91%± 3.12%
Medium (S=250)	Original	41.52%±2.97%	42.10%±3.90%	41.76%±3.32%
	Rules*	**70.07%± 1.7%**	**75.77%± 1.9%**	**72.8%± 1.54%**
	Paraphrased	67.34%± 1.03%	69.51%± 2.02	68.38%± 1.32%
	Article	66.50%± 1.20%	66.34%± 1.66%	66.41%± 1.34%
	One Tag	56.21%± 1.53%	47.97%± 2.70%	51.72%± 2.18%
	Unique Tag	56.70%± 2.34%	49.23%± 3.24%	52.68%± 2.84%
Large (S=500)	Original	65.93%±1.05%	70.49%±1.45%	68.11%±0.83%
	Rules*	**75.91%± 1.76%**	**82.19%± 1.21%**	**78.92%± 1.34%**
	Paraphrased	74.41%± 1.20%	79.14%± 0.85	76.70%± 0.99%
	Article	72.75%± 0.84%	75.92%± 0.47%	74.30%± 0.63%
	One Tag	68.58%± 1.02%	67.45%± 1.25%	68.00%± 0.96%
	Unique Tag	69.96%± 0.88%	69.91%± 1.40%	69.91%± 0.94%

*Label-Wise Token Replacement, 10% Rate. The "Original" method is the un-augmented data. "Rules" and "Paraphrased" are using data augmented with the respective techniques. The "Article" method uses summarization on un-adjusted source text while "One Tag" and "Unique Tag" apply summarization on the adjusted source texts as discussed in Table V.

[2] B. Ding, L. Liu, L. Bing, C. Kruengkrai, T. H. Nguyen, S. R. Joty, L. Si, and C. Miao, "DAGA: data augmentation with a generation approach for low-resource tagging tasks," *CoRR*, vol. abs/2011.01549, 2020. [Online]. Available: https://arxiv.org/abs/2011.01549

[3] R. Wang and R. Henao, "Unsupervised paraphrasing consistency training for low resource named entity recognition," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, M. Moens, X. Huang, L. Specia, and S. W. Yih, Eds. Association for Computational Linguistics, 2021, pp. 5303–5308. [Online]. Available: https://doi.org/10.18653/v1/2021.emnlp-main.430

[4] U. Yaseen and S. Langer, "Data augmentation for low-resource named entity recognition using backtranslation," *CoRR*, vol. abs/2108.11703, 2021. [Online]. Available: https://arxiv.org/abs/2108.11703

[5] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, Q. Liu and D. Schlangen, Eds. Association for Computational Linguistics, 2020, pp. 38–45. [Online]. Available: https://doi.org/10.18653/v1/2020.emnlp-demos.6

[6] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault,

Eds. Association for Computational Linguistics, 2020, pp. 7871–7880. [Online]. Available: https://doi.org/10.18653/v1/2020.acl-main.703

[7] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *J. Mach. Learn. Res.*, vol. 21, pp. 140:1–140:67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[8] J. Zhang, Y. Zhao, M. Saleh, and P. J. Liu, "PEGASUS: pre-training with extracted gap-sentences for abstractive summarization," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 11 328–11 339. [Online]. Available: http://proceedings.mlr.press/v119/zhang20ae.html

[9] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[10] J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran, "Learning multilingual named entity recognition from wikipedia," *Artif. Intell.*, vol. 194, pp. 151–175, 2013. [Online]. Available: https://doi.org/10.1016/j.artint.2012.03.006

[11] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, and T. Solorio, Eds. Association for Computational Linguistics, 2019, pp. 4171–4186. [Online]. Available: https://doi.org/10.18653/v1/n19-1423