Abstract
In this paper, we leverage pre-trained language models (PLMs) to precisely evaluate the semantics preservation of edition process on sentences. Our metric, Neighboring Distribution Divergence (NDD), evaluates the disturbance on predicted distribution of neighboring words from mask language model (MLM). NDD is capable of detecting precise changes in semantics which are easily ignored by text similarity. By exploiting the property of NDD, we implement an unsupervised and even training-free algorithm for extractive sentence compression. We show that NDD-based algorithm outperforms previous perplexity-based unsupervised algorithm by a large margin. For further exploration on interpretability, we evaluate NDD by pruning on syntactic dependency treebanks and apply NDD for predicate detection as well.

1 Introduction
Sentence editions, like deletion and replacement (Liu et al., 2020; Huang et al., 2021; Xu and Durrett, 2019a), are widely used in natural language processing (NLP) to complete generative tasks in an extractive procedure. Many such tasks require model to maintain most semantics, including text compression and rewriting. However, metrics for semantics comparison remain insufficient. Perplexity emphasizes more on structural integrity rather than semantics and text similarity is not precise enough for a satisfying performance.

As the two cases in Figure 1, we execute an edition (replacement) for each sentence. In the first case, we keep the semantics almost unchanged while in the second case, the replacement from river into town obvious leads to a semantics change, especially for the meaning of bank. However, conventional cosine similarity fails to capture the semantics shifting in the second case as it predicts a similarity close to the first case.

Thus, we introduce our novel metric, Neighboring Distribution Divergence, to precisely detect the semantics changes caused by text edition. NDD evaluates based on pre-trained language models like BERT (Devlin et al., 2019). NDD is designed based on the assumption that changes in semantics can be reflected by predicted distribution changes of neighboring words. For instance, when we use masked language model to predict the masked bank in The man sits beside the bank of the river, words like source or surface will more likely be predicted. If we replace river by bank, which leads to a semantics change, the probability of words like center or college to be predicted will become higher. In contrast, if river is replaced by lake, source be surface will still be predicted with high confidence, which indicates the edition preserves the initial semantics.

In Specific, NDD predicts distributions of masked neighboring words before and after the edition. Then these distributions are calculated by the KL divergence function and summed up to get the final metric. A higher NDD indicates greater change in semantics of a sentence. As shown in Figure 1, edition in the second case results on more than $\times 2.5$ NDD than the first case, which reflects high precision of NDD’s detection on the semantics change.

Based on NDD’s property, we use this metric to detect semantics changes during text compression.
and implement a NDD-based compressing algorithm. For each compressing step, we evaluate the NDD caused by this edition and only allow the edition when NDD is under the threshold. As the procedure is totally guided by a fixed PLM, our algorithm is unsupervised and free from any training on large corpus for compression. With the comparison between perplexity-based algorithm, NDD-based algorithm outperforms by a large margin and is shown to be much more capable of preserving semantics.

We conduct experiments on syntactic and semantic treebanks to explore NDD’s awareness of syntax and semantics. To be specific, NDD enables unsupervised pruning on syntactic dependency treebanks and predicate mining. This shows NDD’s awareness of syntax and semantics without training on related datasets, which verifies the potential of NDD on more NLP tasks. Our contributions can be concluded as follows:

- We propose a novel PLM-based metric NDD to evaluate the semantics preservation or change caused by text edition.
- We implement a NDD-based training-free algorithm which performance significantly better than previous perplexity-based algorithm on unsupervised text compression.
- Our further experiments on syntactic and semantic treebanks show NDD’s awareness of syntax and semantics.

2 Neighboring Distribution Divergence

In this section, we give an elaborate description of the procedure to calculate the NDD metric. In NDD, distribution refers to the predicted probability distributions in MLM, divergence refers to the KL divergence of the predicted distributions before and after the edition, and neighboring means that more attention will be paid to words near the edited spans. NDD directly reflects the semantic disturbance on other unedited words caused by the edition.

Given a sentence $W$ with $n$ words $W = [w_1, w_2, \cdots, w_n]$, an editing operation $E$ is used to convert the sentence to an edited one. For formula simplification, we suppose $E$ to be a replacement for discussion. Suppose that $E$ replaces a span $[w_i, w_{i+1}, \cdots, w_j]$ in $W$ with a span $V = [v_1, v_2, \cdots, v_k]$, then the new sentence will be $W' = [w_1, \cdots, w_{i-1}, v_1, v_2, \cdots, v_k, w_{j+1}, \cdots, w_n]$. Then we calculate the predicted distribution divergence on those neighboring words $[w_1, \cdots, w_{i-1}, w_{j+1}, \cdots, w_n]$ of the edition. We use MLM-based prediction as depicted in Figure 2.

For a sentence $W$, we predict the MLM-based distribution on $i$-th position as follows.
$W_m = [w_1, \cdots, w_{i-1}, [\text{MASK}], w_{i+1}, \cdots, w_n]$.

$R = \text{PLM}(W_m); d = \text{softmax}(R_i) \in \mathbb{R}^c$

We first mask the word on $i$-th position and then apply PLM for prediction on that position. Finally, a softmax function is used to get the probability distribution $D$ where $d_j$ refers to the appearance possibility of $j$-th word in the c-word dictionary on $i$-th position. We summarize this distribution predicting process with a function $\text{MLM}(\cdot)$ where $\text{MLM}(W, i) = d$.

Then we go back to the discussion of text edit. For the edition $E$, we use $\text{MLM}(\cdot)$ to predict the distribution $D = [d_1, \cdots, d_{i-1}, d_{j+1}, \cdots, d_n]$ of neighboring words in the unedited sentence $W$. We calculate another distribution $D'$ for neighboring words in the edited sentence $W'$.

After we get the distributions $D$ and $D'$, we use KL divergence to calculate the difference between the two distributions.

$$\text{div} = D_{KL}(d'||d) = \sum_{i=1}^{c} d'_i \log(\frac{d'_i}{d_i})$$

Here we use $D$ from the unedited sentence as the observed distribution and $D'$ from the edited sentence for approximation. After we get the divergence $\text{div}$ between each pair in $D$ and $D'$, we use a weighted sum for the final NDD score.

$$\text{NDD}(W, W') = \sum_{k \in [1, \cdots, i-1, j+1, \cdots, n]} a_k \text{div}_k$$

where $a_k$ is the distance weight which can be designed as $\mu^\text{min}(|k-i|,|k-j|) (\mu \leq 1.0)$. Distance weight is added for scaling that more divergence will be detected on words closer to edited spans. In latter experiments, this weight will be re-designed for specific tasks, but generally, words closer to edited spans will be assigned higher weights.

So why is NDD capable to capture precise semantics changes? First, NDD uses predicted distributions to represent words rather than just the word itself. As shown in Figure 2, the disturbance on semantics is not just detected by existing words like be and mine, but is detected by all words in the dictionary as well. Moreover, this procedure enables PLM to evaluate semantic disturbance on unknown words much better. For instance, if a PLM meets an unknown word like Okinawa, it will use it as an [UNK] token to calculate the perplexity. In contrast, we replace evaluation directly on real words by evaluating on appearance likelihood. Thus, the PLM will be able to know this word to have like 10% probability to be place, 20% probability to be region, etc. from the surrounding words. Finally, NDD compares the semantics between sentences before and after edit, which is unlikely to be implemented using perplexity. Perplexity can only be used to evaluate the fluency of the edited sentence while NDD is also able to detect whether the semantics has been preserved.

### 3 Evaluating Precise Semantic Similarity

Following the discussion in the introduction, we use cases to further explore the ability of our model to capture precise semantics changes using several examples. As in Table 1, we edit the initial sentence “I am walking in the cold rain.” with a series of replacement. We keep syntactic structure of the sentence unchanged and replace some words by other words with the same POS. Thus, the difference between the initial and edited sentences is majorly the semantics.

We divide the editing cases into several groups. In the first three groups, we change words (adjective, noun and verb respectively) into similar, different or opposite meanings. NDD successfully detects the semantics changes and is able to precisely evaluate the changing extents. Taking the first group as the instance, changing from cold into cool and freezing keeps the most semantics while changing into hot leads to the opposite and even

<table>
<thead>
<tr>
<th>Sentence</th>
<th>PPL</th>
<th>NDD</th>
<th>Cos.Sim.</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am walking in the cold rain.</td>
<td>5.99</td>
<td>0.00</td>
<td>1.000</td>
</tr>
<tr>
<td>I am walking in the cool rain.</td>
<td>10.10</td>
<td>0.81</td>
<td>0.995</td>
</tr>
<tr>
<td>I am walking in the freezing rain.</td>
<td>5.63</td>
<td>0.97</td>
<td>0.997</td>
</tr>
<tr>
<td>I am walking in the heavy rain.</td>
<td>5.30</td>
<td>1.82</td>
<td>0.994</td>
</tr>
<tr>
<td>I am walking in the hot rain.</td>
<td>14.77</td>
<td>3.17</td>
<td>0.995</td>
</tr>
<tr>
<td>I am walking in the cold snow.</td>
<td>5.37</td>
<td>2.46</td>
<td>0.996</td>
</tr>
<tr>
<td>I am walking in the cold night.</td>
<td>6.18</td>
<td>3.52</td>
<td>0.991</td>
</tr>
<tr>
<td>I am walking in the cold sunshine.</td>
<td>8.59</td>
<td>4.73</td>
<td>0.994</td>
</tr>
<tr>
<td>I am running in the cold rain.</td>
<td>11.86</td>
<td>0.66</td>
<td>0.990</td>
</tr>
<tr>
<td>I am wandering in the cold rain.</td>
<td>16.89</td>
<td>0.89</td>
<td>0.982</td>
</tr>
<tr>
<td>I am swimming in the cold rain.</td>
<td>14.84</td>
<td>3.29</td>
<td>0.986</td>
</tr>
<tr>
<td>He is walking in the cold rain.</td>
<td>10.32</td>
<td>4.72</td>
<td>0.980</td>
</tr>
<tr>
<td>He is walking in the cold rain.</td>
<td>105.55</td>
<td>13.04</td>
<td>0.991</td>
</tr>
<tr>
<td>He is walking in the cold rain.</td>
<td>13.95</td>
<td>7.22</td>
<td>0.980</td>
</tr>
</tbody>
</table>

Table 1: Cases for detection of NDD on very precise semantics changes. The initial sentence is “I am walking in the cold rain.”
implausible semantics. NDD reflects the difference of semantics between these edited results and assigns a much higher score to the cold-to-hot case. Moreover, in the medium case where the aspect for description is changed to heavy, NDD remarkably assigns a medium score to this case, showing its high discerning capability.

In the last case group, we change the tense and subject of the sentence. NDD is shown to be fairly sensitive to tenses and subjects. This property can be used to retain those critical properties during editions. NDD is also able to detect syntactic faults like the combination of He am and can thus be used for fault preventing.

From these cases we can also see why perplexity and cosine similarity is incapable of detecting precise semantics changes as NDD. In Table 1, cosine similarity cannot detect subtle semantics changes and even syntactic faults. We attribute this to the high reliance on word representations for sentence represents as sentences with many words overlapped will be classified to be similar.

For perplexity, the first problem with it is that this metric evaluates a single sentence rather than a pair of sentences. Thus, perplexity can only estimate the plausibility of sentences instead of semantic relationships. Perplexity will thus guide editions to transform sentences into more syntactically plausible versions. As shown in Table 1, edited results with lower perplexity may change semantics like cold-to-heavy and rain-to-snow. NDD is able to preserve semantics much better by suggesting changing cold to cool or freezing and changing walking to running or wandering.

Another reason is that perplexity can easily be misguided by low-frequency words. In the walking-to-wandering case, the resulted perplexity is even higher than the walking-to-swimming case. Since perplexity is scored based on existence probability of words, the low-frequency wandering will lead to a higher perplexity, even though wandering is semantically closer to walking than swimming. This issue is overcome in NDD as we use predicted distributions rather than real words. As described before, NDD can understand low-frequency words and even named entities much better. As the result, NDD correctly scores the semantics changes caused by replacement on walking.

4 Unsupervised Text Compression
To show the advantages of NDD in application, we implement an unsupervised algorithm for text compression guided by NDD.

4.1 Span Searching
Given a sentence $W$, we try every span $W_{ij} = [w_i, \cdots, w_j]$ with length under a certain limit $L_{\text{max}}$ for deletion. Then we use NDD to score the semantics changes caused by the deletions.

$$S_{ij} = \text{NDD}(W, W_{ij})$$

$$W'_{ij} = [w_1, \cdots, w_{i-1}, w_{j+1}, \cdots, w_n]$$

where all words in $W'$ are used as neighboring words for metric calculating. We select spans with NDD under a certain limit $NDD_{\text{max}}$ as the candidates for the next processing step.

4.2 Overlapped Span Selection
As overlapping often occurs in the spans from searching, we apply a simple selective algorithm to filter the candidate spans. Specifically, we compare each overlapped span pairs, in which two spans contain some common words. For each pair, we delete the span with lower NDD score and keep the other span for next round of comparison. This process iterates until there is no overlapped span in candidates.

4.3 Other Details
As following the distance weights described before are imbalanced for words near the start and end of a sentence. In practice, we use a modified balanced weights for distance.

$$a_k = \mu^{\min(|k-i|,|k-j|)}$$

$$a'_k = a_k + a_{n'-k} * \mu^{n'}$$

$$n' = n - (j - i + 1)$$

The main effect of this modification is to let words near the two side to be detected twice for their disturbance on neighboring words. With the help of this modification, we overcome the weight imbalance issue and thus avoid incorrect deletions.

Furthermore, we add another weight $b_k$ to encourage our algorithm to delete latter words in the sentence as it is less common to use these words for summary. We modify the weighted sum as follows.
Table 2: Results for sentence compression on the Google dataset, we compare our algorithm with other unsupervised algorithms. Underlines mean the improvement to be significant \( (p < 0.05) \) considering the highest baseline. \(*\): Re-implementation

\[
b_k = \nu_k^k
\]

\[
\text{NDD}(W, W') = \sum_{k \in \{1, \ldots, \beta - 1, j+1, \ldots, n\}} a_k' b_k d i v_k
\]

### 4.4 Experiment

**Dataset and Configuration** To compare our algorithm with previous algorithms, we conduct our experiments on the Google dataset (Filippova et al., 2015). We use the evaluation dataset with 10,000 sentence pairs for performance evaluating. We use BERT-base-cased which has been specialized for the MLM task as the PLM. We set \( L_{\max} \) to 9 and \( NDD_{\max} \) to 1.0 for span filtering. For weighting, we set \( \mu \) and \( \nu \) both to 0.9. Compressing rate is controlled under 0.6 as in previous works. We choose BLEU (Papineni et al., 2002) and F1 score as metrics for evaluation and comparison because precision is more critical than recall (Returning the whole unedited sentence results in a high recall) in extractive compression.

**Results** Our results are shown in Table 2, we report the result from (Niu et al., 2019) and re-implement the claimed PPL-based algorithm. We find our implementation performs a little higher than the reported result. However, the result is still poor and even far from the unedited baseline. Our compression algorithm significantly outperforms the PPL-based algorithm by 17.4 F1 score on unsupervised sentence compression. For further exploration, we randomize our algorithm by deleting random words of the same number as in NDD-based algorithm for each sentence. Results in Table 2 show that PPL-based algorithm even does not have a significant improvement comparing with the randomized algorithm. This implies that only keeping the fluency of sentences by considering perplexity does not help much for sentence compression. In contrast, NDD has the ability to guide the algorithm to remove subordinated components by preserves semantics in each edition step. Thus, NDD performs much better than perplexity on sentence compression to produce semantics preserved output.

Comparing with supervised methods, our algorithm still has a long way to go. But we will show in the next sections that automatic metrics are biased for evaluating the performance of our compression as difference exists in compressing styles between outputs from unsupervised compression and annotated gold results.

**Analysis of automatic metrics**

Table 3: Examples for how automatic metrics reflect the performance of NDD-based compression. Improvement refers to comparison with unedited texts.

<table>
<thead>
<tr>
<th>Method</th>
<th>F1</th>
<th>B1</th>
<th>B2</th>
<th>B3</th>
<th>B4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Unsupervised)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Unedited</td>
<td>63.2</td>
<td>44.8</td>
<td>34.9</td>
<td>28.3</td>
<td>23.5</td>
</tr>
<tr>
<td>Random</td>
<td>45.7</td>
<td>43.0</td>
<td>25.4</td>
<td>16.2</td>
<td>10.4</td>
</tr>
<tr>
<td>PPL-based (Niu et al., 2019)</td>
<td>50.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PPL-based*</td>
<td>52.3</td>
<td>45.9</td>
<td>35.5</td>
<td>19.9</td>
<td>14.7</td>
</tr>
<tr>
<td>NDD-based (ours)</td>
<td>67.4</td>
<td>54.8</td>
<td>39.3</td>
<td>30.5</td>
<td>23.7</td>
</tr>
</tbody>
</table>

### 4.5 Compression Cases

**Real Effect vs. Automatic Metrics** As the compressed results for sentences can be various, automatic metrics might not be able to fully reflect the compressing ability of our algorithm. Also, as our compression follows a training-free procedure, the compressed results might not be in the same style as the annotated golden ones like the first instances in Table 3. Both our compressed and the golden result keep the main point that the speed limit will be 70 mphs, preserving the semantics of the whole sentence. However, the golden compression tends to keep some auxiliary information like the location on highways in Illinois and the time next year. In contrast, NDD-based compression tends to remove those unimportant information and prevent
Init: A US$5 million fish feed mill with an installed capacity of 24,000 metric tonnes has been inaugurated at Prampram, near Tema, to help boost the aquaculture sector of the country.

Iter1: A US$5 million fish feed mill with an installed capacity of 24,000 metric tonnes has been inaugurated at Prampram, near Tema, to help boost the aquaculture sector of the country.

Iter2: A fish feed mill with capacity 24,000 has been inaugurated at Prampram to boost the aquaculture sector.

Final: A mill has been inaugurated to boost aquaculture sector.

Table 4: Cases for output in iterations of the NDD-based compression. Bold: Kept components

Outputs from Compression Iterations We present the intermediate outputs of our algorithm in Table 4. NDD-based text compression is shown to be capable of detect and remove auxiliary components like locations or adjective spans in the sentence for example. Also, the syntactic integrity and initial semantics are preserved in each iteration of our algorithm. There is an advantage over supervised methods as output in each iteration is still a plausible compression for the initial sentence. We can thus set some proper thresholds and iterate the compression until we get a fully satisfying output.

Compression on Other Languages We also implement algorithm for other languages to verify the cross-lingual capability of NDD-based compressing. Cases in Table 5 show our algorithm to be pretty well-performed on compression of other languages.

5 Syntactic Dependency Tree Pruning

We further analyze our metric and algorithm on upstream tasks. To show that NDD understands semantics in other parts of the sentence to be unchanged. Thus, NDD-based compression still keep despite opposition from organizations towards the integrated semantics. In the second instance of Table 3, as the golden compression also remove location and time information from the sentence, our algorithm leads to a significant improvement since our compressing style matches with the annotated one. Considering that the automatic metrics may be biased due to the style of annotation, we present more cases in this section to show the capacity of our algorithm to keep semantics and fluency while removing unimportant and auxiliary components at the same time.

Semantics on Other Languages Our algorithm captures the core linguistic information from the sentence, which helps it to make a better decision on which component to remove. We observe that NDD-based compression is more capable of keeping the core semantics than the word- and span-level techniques. The second advantage is its ability to preserve the syntactic structure. Higher Depth-n for larger n and lower Depth-n for smaller n indicates better preservation of the syntactic structure. Higher Subtree-n indicates the pruned spans result in less damage to the syntactic integrity.

We experiment on the test data of PTB-3.0 dataset (Marcus et al., 1993). We randomize our
algorithm as before for a fair comparison with the same compressing rate. For our algorithm in different configuration, we implement a corresponding randomized algorithm for preciser comparison. As in Table 6, the awareness of syntax is verified for both node and span pruning. First, the proportion of nodes in shallower levels (depth=1 ~ 3) pruned by our algorithm are smaller than all the corresponding proportion when pruned nodes are randomized. NDD-based pruning is more likely to prune deeper nodes (depth ≥ 4) in the syntactic dependency tree. Also, the proportion of subtrees in spans pruned by NDD-based algorithm is significantly larger (30 ~ 50) than the randomized correspondents. Thus, we conclude that NDD is able to guide the compressing algorithm to detect subordinated components in syntax dependency treebanks even though the PLM has never been trained on any syntactic datasets.

For comparison among different configurations, a lower $NDD_{\text{max}}$ will lead both node and span pruning to improve. This is natural as the lower threshold will only allow the algorithm to prune components with little disturbance to semantics. For $L_{\text{max}}$, when $NDD_{\text{max}}$ is low, a higher $L_{\text{max}}$ will improve the node pruning by pruning more auxiliary components in deeper levels. For instance, long spans like since the early 1970s in Figure 3 might not be detected when $L_{\text{max}}$ is low. But for a higher $NDD_{\text{max}}$, $L_{\text{max}}$ will lead to higher proportion of subtrees in pruned spans as higher $L_{\text{max}}$ may allow longer spans which are not subtrees to be pruned.

### 6 Predicate Detection

As pruning on the syntax dependency treebanks shows NDD to have the understanding of syntax, we further explore the discerning ability of NDD for semantic components on large datasets. We choose to experiment on the semantic role labeling (SRL) dataset for predicate detection. In the experiment, words in the sentence are edited by deletion or replacement and semantics changes caused by these editions are evaluated using NDD. As predicates are semantically related to more components (augments) in the sentence, higher NDD refers to higher probability of an edited word to be a predicate. Thus, we evaluate the predicate detecting ability following with the words ranking task. We rank the probability of words to be predicates and use ranking metrics mean average precision (mAP) and area under curve (AUC) for evaluation.

We conduct our experiments on Conll09 SRL datasets (Hajic et al., 2009). To test the generalizing ability of our method, we experiment on both in-domain (ID) and out-of-domain (OOD) English (ENG) datasets. Another Spanish (SPA) dataset is also involved for cross-language evaluation. We edit each word in the sentence in three ways: (a) Directly delete the word, (b) Replace the word with a mask token, (c) Replace the word with a certain word (a for ENG-ID, that for ENG-OOD and el for SPA). We apply SpanBERT-base-cased (Joshi et al., 2020) and BERT-base-spanish-cased (Cañete et al., 2020) as PLMs. For comparison, we also implement a PPL-based algorithm which likewise uses perplexity to determine predicates.

Our main results are presented in Table 7, showing that PPL might not be a proper metric to detect predicates as AUCs that result from PPL-based algorithm are around 40 ~ 60 and mAPs are generally poor. In contrast, NDD-based algorithm produces much better results and outperforms PPL-based algorithm by 10 ~ 20 scores on both AUC and mAP metrics, which is a remarkably significant margin and verifies NDD to be much more capable in understanding semantics. We also ensemble the three editions by using the product of three predicted probabilities. The ensemble algorithm leads to further improvement and lifts AUC, mAP to higher than 80.0, 50.0 respectively, even making it a plausible way to detect predicates following an
unsupervised procedure.

Comparison among editions shows that direct deletion will lead to the better performance than other editions evaluated by AUC. Replacing with a certain word perform better on ENG-ID and SPA when we evaluate algorithms with the mAP. Thus, we conclude that deleting predicates causes the greatest disturbance on other components (augments) in the sentence and makes the disturbance more prominent for our algorithm to detect. Also, as a, that and el may empirically outperform other words when being used to detect predicates, those words with low semantic meanings might be advisable choices for predicate detection using word replacement.

### 7 Related Works

Text similarity and perplexity are metrics which can be used for many downstream tasks (Park et al., 2020; Lakshmi and Baskar, 2021; Nguyen-Son et al., 2021; Campos et al., 2018; Neishabouri and Desmarais, 2020; Lee et al., 2021). Unfortunately, these metrics are not precise enough to detect semantics changes as discussed before. Recent study (Kuribayashi et al., 2021) shows that low perplexity does not directly refer to a human-like sentence. Therefore, we should consider again how to evaluate subtle text difference like semantics shift caused by an edition on the text.

Therefore, we assume PLM like BERT (Devlin et al., 2019) to be a chance for some changes. PLM-based metrics like BERT score has been verified by experiments to evaluate text generation better (Zhang et al., 2020). Instead of matching words exactly, BERT score computes pairwise cosine similarity between words in texts and use greedy matching for the final scoring. Our NDD also puts real words aside but uses distributions predicted from MLM to represent words. We use KL divergence to estimate the semantic difference between texts. Other works are also pursuing better metrics than strict matching scores like BLEU for generative tasks. To evaluate semantics preservation in AMR-to-sentence, (Opitz and Frank, 2021) exploits pre-trained AMR parser to compare the AMR graph of generated results with the golden graph, showing the potential of pre-trained model in evaluation.

Sentence compression is currently dominated by supervised methods (Malireddy et al., 2020; Nguyen et al., 2020; Nóbrega et al., 2020) and highly relies on syntactic dependency trees (Le et al., 2019; Xu and Durrett, 2019b; Wang and Chen, 2019; Kamigaito and Okumura, 2020). Un-supervised methods have been explored to extract sentences from documents to represent the key points (Jang and Kang, 2021). But the performance on pruning components in sentences is still far from satisfaction. (Niu et al., 2019) explores evaluating the perplexity of outputs after compression. Comparing with NDD, such metric is fairly less capable to detect semantics changes in editions and thus cannot preserve the semantics.

Annotated data from parsing tasks like syntactic dependency parsing (Dozat and Manning, 2017; Li et al., 2020b) and semantic role labeling (Li et al., 2020a,c) can reflect model’s awareness of those internal relationships between words in sentences. Experiments show NDD to perform well on detecting those relationships. Thus, we may explore unsupervised procedures for those tasks based on NDD in the future.

### 8 Conclusion

In this paper, we propose a novel metric, neighboring distribution divergence, to evaluate very precise semantics changes caused by editions. We implement an unsupervised and training-free algorithm for text compression and find that NDD-based algorithm outperforms PPL-based algorithm by a large margin. Also, NDD-based text compression can still produce highly semantics-preserved outputs even when human-annotated data cause automatic metrics to be biased. We further explore for whether NDD has a real awareness of semantics and verify our hypothesis as NDD perform well on pruning components in sentences from documents to represent the key points (Jang and Kang, 2021). But the performance on pruning components in sentences is still far from satisfaction. (Niu et al., 2019) explores evaluating the perplexity of outputs after compression. Comparing with NDD, such metric is fairly less capable to detect semantics changes in editions and thus cannot preserve the semantics.

### Table 7: Evaluation on ability of metrics to detect predicates in sentences.

<table>
<thead>
<tr>
<th>Edition</th>
<th>ENG-ID</th>
<th>ENG-OOD</th>
<th>SPA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mAP</td>
<td>AUC</td>
<td>mAP</td>
</tr>
<tr>
<td>(NDD-based)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>52.1</td>
<td>75.8</td>
<td>61.1</td>
</tr>
<tr>
<td>Replace by mask</td>
<td>48.2</td>
<td>74.6</td>
<td>56.8</td>
</tr>
<tr>
<td>Replace by word</td>
<td>51.6</td>
<td>77.2</td>
<td>56.2</td>
</tr>
<tr>
<td>Ensemble</td>
<td>54.3</td>
<td>80.0</td>
<td>63.7</td>
</tr>
<tr>
<td>(PPL-based)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>36.8</td>
<td>56.8</td>
<td>44.5</td>
</tr>
<tr>
<td>Replace by mask</td>
<td>35.9</td>
<td>56.7</td>
<td>33.1</td>
</tr>
</tbody>
</table>

Sentence compression is currently dominated by supervised methods (Malireddy et al., 2020; Nguyen et al., 2020; Nóbrega et al., 2020) and highly relies on syntactic dependency trees (Le et al., 2019; Xu and Durrett, 2019b; Wang and Chen, 2019; Kamigaito and Okumura, 2020). Un-supervised methods have been explored to extract sentences from documents to represent the key points (Jang and Kang, 2021). But the performance on pruning components in sentences is still far from satisfaction. (Niu et al., 2019) explores evaluating the perplexity of outputs after compression. Comparing with NDD, such metric is fairly less capable to detect semantics changes in editions and thus cannot preserve the semantics.

Annotated data from parsing tasks like syntactic dependency parsing (Dozat and Manning, 2017; Li et al., 2020b) and semantic role labeling (Li et al., 2020a,c) can reflect model’s awareness of those internal relationships between words in sentences. Experiments show NDD to perform well on detecting those relationships. Thus, we may explore unsupervised procedures for those tasks based on NDD in the future.
References


A Appendix

A.1 Case Translation

<table>
<thead>
<tr>
<th>Init</th>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>调价周期内，沙特下调10月售往亚洲的原油价格，我国计划释放储备原油，油价一度承压下跌。 (Translation) During the price adjustment, Saudi scales down</td>
<td>(Translation) During the price adjustment, Saudi scales down the price of crude oil sold to Asia in October, our country plans to release the reserved crude oil, oil price has once been under the dropping pressure.</td>
</tr>
<tr>
<td>我国计划释放储备原油，油价一度承压下跌。 (Translation) During the price adjustment, Saudi scales down the price of crude oil, oil price has once been under the dropping pressure.</td>
<td>our country plans to release the reserved crude oil, oil price has once been under the dropping pressure.</td>
</tr>
<tr>
<td>El comité de crisis, aseguró el presidente, ha tomado decisiones estratégicas que, por seguridad, no pueden ser reveladas pero que serán evidentes en las acciones que se ejecutarán en las próximas horas. (Translation) The crisis committee, the president assured, has made strategic decisions that, for security, cannot be disclosed but which will be evident in the actions that will be carried out in the next few hours.</td>
<td>El comité de crisis ha tomado decisiones que no pueden ser reveladas pero serán evidentes en las acciones que se ejecutarán. (Translation) The crisis committee has made decisions that cannot be disclosed but will be evident in the actions to be carried out.</td>
</tr>
<tr>
<td>大型で非常に強い台風16号は、10月1日の明け方以降、非常に強い勢力で伊豆諸島にかなり近づく見込みです。 (Translation) Very strong typhoon No.16 with a large scale is expected to closely approach to the Izu Islands with a very strong force after the dawn of October 1.</td>
<td>(Translation) Typhoon No.16 is expected to approach to the Izu Islands.</td>
</tr>
</tbody>
</table>

Table 8: Translation for cases in Table 5.