Appendix

Training Details

For the experiments before the **Final Performance** section, we conduct them under the same setting for a fair comparison. Specifically, we initialize a BERT (Devlin et al. 2019) model from the pre-trained `bert-base-uncased` checkpoint and use the last hidden state of the `[CLS]` token as the sentence embedding. We train the model under the hyper-parameters listed in Table 5. We run trainings on the Wiki data domain for one epoch, and on the NLI data domain for three epochs. We record the Spearman's correlation on the evaluation data, i.e., the validation split of STS Benchmark dataset, as well as the alignment and uniformity of both held-out training data and evaluation data, every 125 training steps. We randomly shuffle the original training data and hold out the top 10% of the total data as the held-out training data. In Figure 1, the final score is the average of top 5 Spearman's correlation.

Optimizer	Learning rate	Batch size	Temp
AdamW	3e-5	256	5e-2

Table 5: Hyper-parameters for experiments before the **Final Performance** section. AdamW is proposed by (Loshchilov and Hutter 2019) and Temp is the temperature τ for contrastive loss.

For the experiments in the **Final Performance** sections, we adopt the pooling strategy from (Jiang et al. 2022) for the sentence embedding to achieve better and more stable performance. Specifically, the sentence s is placed in the following prompts:

- *This sentence: "s" means [MASK]*;
- *This sentence of "s" means [MASK].*

Then, the last hidden state of `[MASK]` is used as the sentence embedding. To determine the optimal combination of hyper-parameters, we retain the same backbone, optimizer, batch size and temp as the previous experiments, and carry out a grid search on the learning rate $\in \{1\mathrm{e}-5, 3\mathrm{e}-5, 5\mathrm{e}-5\}$, $m_1 \in \{1\mathrm{e}-3, 5\mathrm{e}-3, 1\mathrm{e}-2\}$, $m_2 \in \{1\mathrm{e}-2, 5\mathrm{e}-2, 1\mathrm{e}-1\}$ and $\beta \in \{0.1, 0.5, 1.0\}$ over the evaluation data, with the selected hyper-parameters listed in Table 6.

	Learning rate	m_1	m_2	β
Wiki.STS_HT	1e-5	5e-3	5e-2	1
NLI.STS_HT	1e-5	5e-3	1e-1	1

Table 6: Hyper-parameters for experiments in the **Final Performance** section.

Our code is implemented in `python 3.9.13`, with `pytorch 1.12.1` (Paszke et al. 2019) and `transformers 4.18.0` (Wolf et al. 2019). The experiments are conducted on a single 32G NVIDIA V100 GPU and repeated three times with different random seeds to obtain the final results.

Training Processes of "token_shuffle" and "token_cutoff"

We record the variety of alignment and uniformity during the training process in the **Observation** subsection. In this subsection, we present the results of "NLI.token_shuffle", "NLI.token_cutoff", "Wiki.token_shuffle" and "Wiki.token_cutoff" as shown in Figure 6. Both exhibit better alignment and uniformity in the held-out training data compared to those in the evaluation data, which is similar to the training process of "NLI.dropout" and "Wiki.dropout" respectively.

Match Error Rate and Cosine Similarity

RFD measures the complexity of the similarity pattern of training data. To intuitively understand the concept of **pattern complexity**, we select two metrics, Match Error Rate (MER) (Morris, Maier, and Green 2004) and Cosine Similarity (CS), to reflect the patterns in the data at both the lexical and feature levels.

Given two sentences s_1 and s_2, and a sentence encoder f that maps each sentence into an embedding in the hypersphere, the two metrics are defined as follows:

- **Match Error Rate** (MER): this metric quantifies the similarity between two sentences based on edit distance. In calculating this distance, there are four permissible operations: insertion, deletion, substitution, and retain. We use I, D, S, R to represent the count of these four operations, respectively.

$$\mathrm{MER}(s_1, s_2) = \frac{I + D + S}{I + D + S + R}. \tag{8}$$

  This metric is calculated at a lexical level, and a lower value means higher similarity;
- **Cosine Similarity** (CS): Cosine similarity, which is assessed in the feature space, is defined as:

$$\mathrm{CS}(s_1, s_2) = f(s_1)^\top f(s_2), \tag{9}$$

  where $\mathrm{BERT}_{\mathrm{base}}$ is utilized as the sentence encoder in our implementation. This metric is calculated at a feature level, and a higher value means higher similarity.

We calculate the MER and CS for both positive and negative sentence pairs in the Wikipedia and NLI datasets under all settings. Additionally, we calculate the MER and CS for the STS Benchmark evaluation dataset (STSB for short), where sentence pairs with an STS score higher than 4 are treated as positive sentence pairs, and those with an STS score lower than 1 are treated as negative. We use the density histogram to describe the distribution of these two metrics and plot the results in Figure 7. The results are organized into four rows: the first row includes the results for the Wikipedia dataset under three unsupervised settings; the second row includes the results for the NLI training dataset and STSB evaluation dataset, both of which contain the patterns we want to simulate; the third row includes the results for the LLM-generated data whose corpus source is Wikipedia dataset; and the fourth row includes the results for the LLM-generated data whose corpus source is NLI dataset.

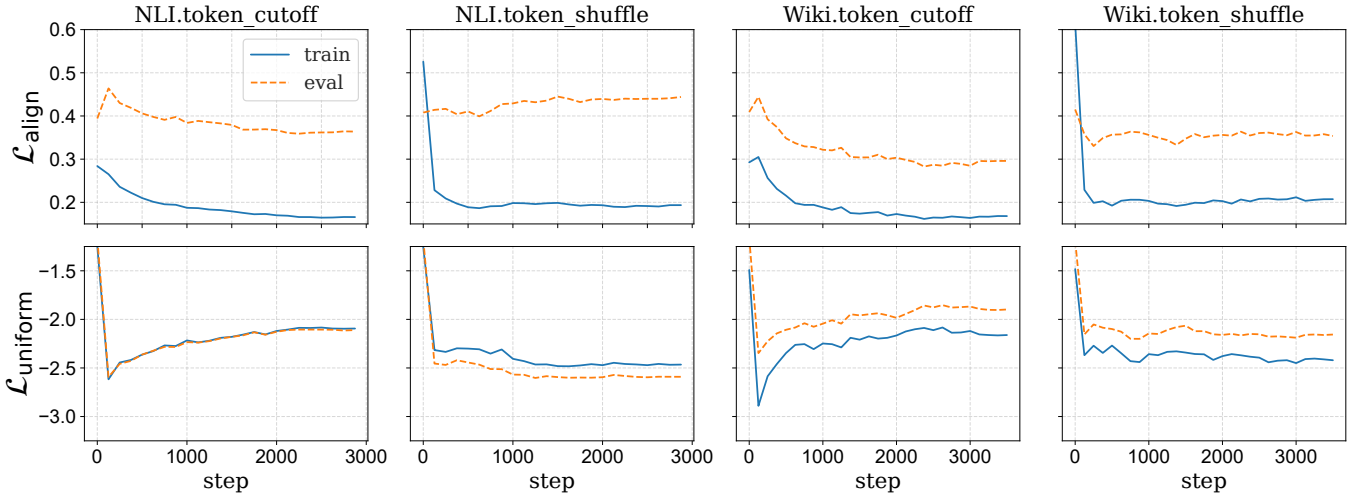

Figure 6: Alignment and uniformity in both the held-out training data and the evaluation data during the training processes of "NLI.token_shuffle", "NLI.token_cutoff", "Wiki.token_shuffle" and "Wiki.token_cutoff".

To intuitively understand the concept of **pattern complexity**, we compare the first two rows. It can be found that the distribution of MER in the supervised setting ("NLI") is broader for both positive and negative sentence pairs compared to that in the unsupervised settings. This implies that the supervised setting contains more complex patterns than the unsupervised settings. Moreover, the distribution of CS in the supervised setting is shifted to the right relative to that in the unsupervised settings for negative sentence pairs, indicating that the negative sentence pairs in the supervised setting are more difficult to identify. The same holds true for positive sentence pairs in most cases. These observations prove that the patterns in the supervised setting are more complex than those in the unsupervised settings, making the patterns in the supervised setting more difficult to fit. It's important to note that RFD is calculated on the same evaluation data. Therefore, when the pattern complexity of the supervised setting is higher than that of the unsupervised setting, the RFD between the supervised setting and the evaluation data is correspondingly higher than the RFD between the unsupervised setting and the evaluation data.

We can also use the MER and CS to intuitively understand the concept of **pattern simulation**. First, we compare the last two rows with the first row. From the comparison, we can observe that the patterns in the LLM-generated data are more complex than those in the unsupervised settings, confirming that we successfully introduce complex patterns in the generated hybrid datasets. Then, we compare the last two rows with the second row and find that the distribution of data generated by simulating STS patterns ("Wiki.STS" and "NLI.STS") aligns more closely with that of "STSB", and the distribution of data generated by simulating NLI patterns ("Wiki.NLI" and "NLI.NLI") aligns more closely with that of "NLI". These findings verify that the complex patterns are indeed introduced through pattern simulation.

Details of the Prompt

We adopt the ICL capability of the LLM to simulate the NLI patterns and STS patterns separately. In this subsection, we provide the specific prompts we use:

Simulating NLI patterns includes two prompts:

1. *Generating Positive Sentence* s_i^{P}:

```
When the user enters a premise text, plea-
se generate a hypothesis text that stands
in an entailment relationship to the given
premise.

In the following illustrative examples,
the Hypothesis is a logical entailment of
the Premise:
- Example 1:
  - Premise: Over the years these bags
    have proved so handy that to keep up
    with demand the local supply of genu-
    ine handmade bags has been augmented
    with imported goods.
  - Hypothesis: These genuine handmade
    bags are very useful and in demand.
- Example 2:
  - Premise: Advocates worry that if subs-
    tantial numbers of welfare mothers are
    pushed into jobs, centers might be
    swamped by demands to serve as many as
    a million added kids.
  - Hypothesis: Centers are only capable
    of serving a limited number of kids.
- Example 3:
  - Premise: A black and white dog is run-
    ning through a snowy field.
  - Hypothesis: A black and white dog is
    outside.

Generate the hypothesis directly without
any other interpretation. The generated
hypothesis should be a logical inference
from the information available in the pre-
```

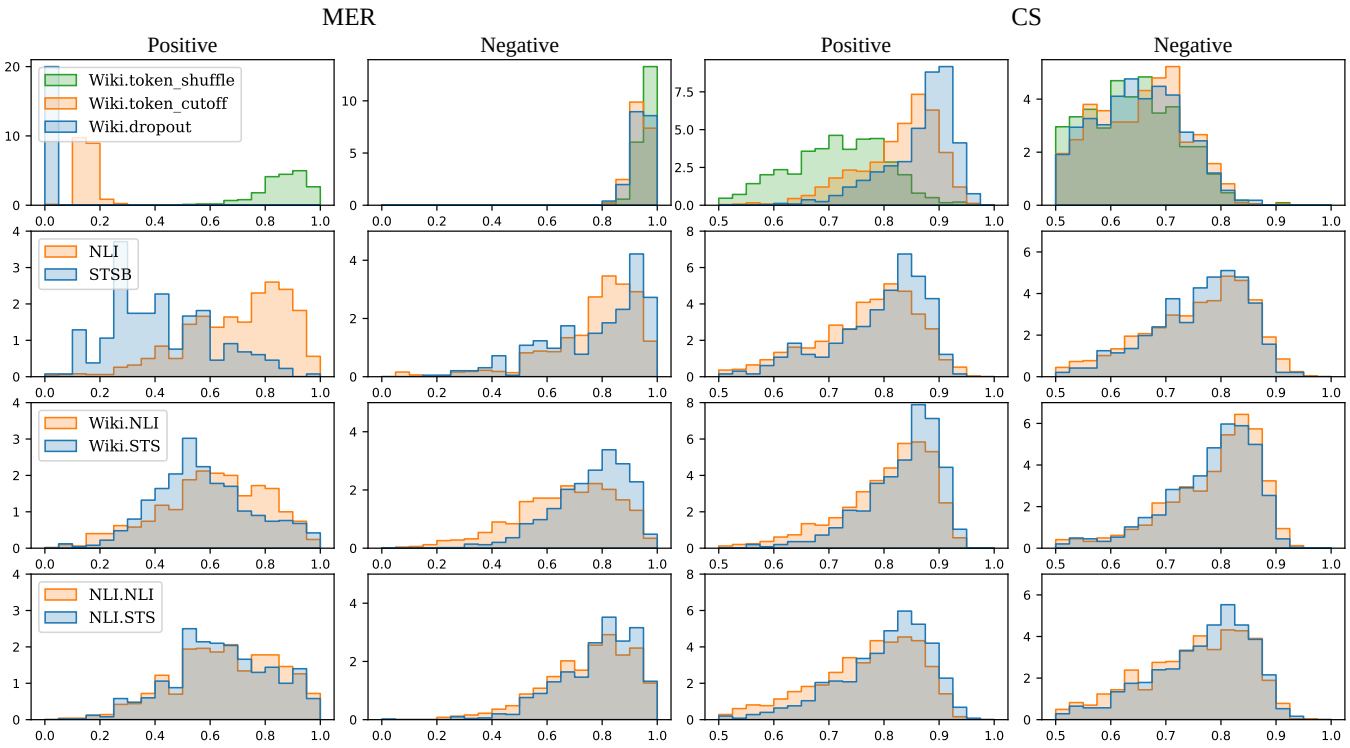

Figure 7: The density histograms of MER and CS for positive and negative sentence pairs in the training data (Wikipedia and NLI) and the evaluation data (STSB) under various settings.

mise. In other words, if the premise is true, the hypothesis must also be true.

2. *Generating Negative Sentence* s_i^{n}:

When the user enters a premise text, please generate a hypothesis text that presents a contradiction to the information provided in the given premise.

In the following illustrative examples, The hypothesis logically contradicts the Premise:
- Example 1:
  - Premise: Over the years these bags have proved so handy that to keep up with demand the local supply of genuine handmade bags has been augmented with imported goods.
  - Hypothesis: The bags are known for falling apart and people are looking for an alternative.
- Example 2:
  - Premise: Advocates worry that if substantial numbers of welfare mothers are pushed into jobs, centers might be swamped by demands to serve as many as a million added kids.
  - Hypothesis: If welfare mothers get jobs, it decreases the number of kids that centers need to serve.
- Example 3:
  - Premise: A black and white dog is running through a snowy field.
  - Hypothesis: The dog is taking a nap.

Generate the hypothesis directly without any other interpretation. The hypothesis should contradict the information given in the premise. This means the premise and hypothesis cannot both be true at the same time.

Simulating STS patterns includes three prompts:

1. *Generating Positive Sentence* s_i^{p}:

Your task is to generate a new sentence that is semantically similar to the user's input sentence.

In the following illustrative examples, Sentence 1 and Sentence 2 are semantically similar:
- Example 1:
  - Sentence 1: U.S. prosecutors have arrtested more than 130 individuals and have seized more than $17 million in a Continuing crackdown on Internet fraud and abuse.
  - Sentence 2: More than 130 people have been arrested and $17 million worth of property seized in an Internet fraud sweep announced Friday by three U.S. government agencies.
- Example 2:
  - Sentence 1: The hearing occurred a day

```
    after the Pentagon for the first time
    singled out an officer, Dallager, for
    not addressing the scandal.
  - Sentence 2: The hearing came one day
    after the Pentagon for the first time
    singled out an officer - Dallager -
    for failing to address the scandal.
- Example 3: Sentence 2 is generated based
  on Sentence 1.
  - Sentence 1: The Bush administration
    blames Hussein loyalists and foreign
    Muslim militants who have entered Iraq
    to fight U.S. troops for the wave of
    bombings and guerrilla attacks.
  - Sentence 2: The Bush administration
    blames the wave of bombings and guess-
    ill attacks on Saddam loyalists and
    foreign Muslim militants who have
    entered Iraq to fight U.S. troops.

Generate the new sentence directly without
any other interpretation, and make sure it
maintains the same information as the ori-
original input sentence.
```

2. *Generating Intermediate Sentence* s_i^{m}:

```
Your task is to generate a revised sentence-
ce by omitting certain details in the
user's input sentence.

In the following illustrative examples,
Sentence 2 is created by omitting details
from Sentence 1:
- Example 1:
  - Sentence 1: Police launched an inter-
National Hunt for Shevaun Pennington
    after she ran away with a 31-year-old
    Toby Studabaker Saturday.
  - Sentence 2: Shevaun Pennington
    disappeared on Saturday morning after
    arranging to meet 31-year-old Toby
Studebaker.
- Example 2:
  - Sentence 1: In a news release Thurs-
    day, Strayhorn said this was the
    first time a comptroller rejected a
    budget.
  - Sentence 2: Strayhorn said it was the
    first time in Texas history a computer-
    roller had not certified the appropriate-
actions act.
- Example 3:
  - Sentence 1: The Korean Air deal is
    expected to be finalized "in the next
    several weeks," Boeing spokesman Bob
    Saling said.
  - Sentence 2: Boeing said the final
    agreement is expected to be signed
    during the next few weeks.

Generate the revised sentence directly
without any other interpretation, and
Make sure that it contains significantly
fewer details than the original input
sentence.
```

3. *Generating Negative Sentence* s_i^{n}:

```
Your task is to generate a new sentence
that conveys a distinct or even contradict-
tory meaning compared to the user's input
sentence.

In the following illustrative examples,
Sentence 2 is generated to convey distinct-
tor contradictory meaning compared to Sen-
tense 1:
- Example 1:
  - Sentence 1: Monkeypox is usually
    found only in central and western
    Africa.
  - Sentence 2: Prairie dogs, usually
    found in southwestern and western
    states aren't indigenous to Wiscon-
    sin.
- Example 2:
  - Sentence 1: The tech-laced Nasdaq Com-
    posite Index gained 2.90 points, or 0.
    18 percent, to 1,606.87.
  - Sentence 2: At 12:10 p.m. EDT, Cana-
    da's benchmark S&P/TSX composite index
    was up 6.87 points or 0.1 percent to
    6,979.29.
- Example 3:
  - Sentence 1: A man is playing the dr-
    ums.
  - Sentence 2: A woman is slicing some
    leaves.

Generate the new sentence directly without
any other interpretation.
```

Every prompt has three examples randomly sampled from the pattern source, and the examples are fixed for every pattern simulation.