Bitext Processing: Translation, Alignment, and Word Sense Disambiguation

1 Word sense disambiguation

1.1 Brief summary

First,we apply AMUSE to the English source sentences to assign the sense tags for the tokens of sentences. Then,we extract the token sense tags with the gold sense tags. Finally, we compare the output with the gold standard to evaluate the quality.

1.2 Setup Of Your evaluation experiments

We can use a four-step pipeline to set up this experiment.

Step 1: Preprocessing Data

We can find significant tokenization discrepancies between the AMUSE API output and the provided English source tokens. For example, AMUSE API tokenizes ”greenhouse-gas” into ’greenhouse’ , ’-’, ’gas’ , while the english source token is ’greenhouse-gas’. Moreover, the AMUSE API tokenizes ’greenhouse gases’ into ’greenhouse’, ’gases’, while the English source token is ”greenhouse gases”.

To address the issues, we can use the following strategy.

1. Iterating through all English source tokens to identify problematic cases

2. Adding the tokens which can be separated by space or hyphens into the dictionary

3. Adding the tokens which contain the special character such as `&amp;` and `Mr.` into the dictionary

4. Sorting keys of dictionary by length with decending order to avoid partial replacement

5. replacing identified tokens in the English source sentences by the dictionary

Step 2: AMUSE API Integration

The preprocessed data are uploaded to AMUSE API website in JSON format with language specification `'lang': 'EN'`. Then, the API returns the tokenized text which contains the tokens with BabelNet synset IDs.

Step 3: Token Alignment and Extraction

We align the API output data with English source tokens and extract the tokens which have the gold sense tags. Additionally, the gold sense tags should be in English source tokens file.

Step 4: Evaluation Using Gold Standard

We can use `evaluate_wsd.py` to compare our created output with the gold standard annotations in `se13.key.txt`. The evaluation returns a score which provides the assessment of WSD system performance.

1.3 Statistics of the data sets and annotations

The SE13 dataset has 301 sentences and 8329 tokens. Among the tokens, there are 1644 tokens annotated by gold sense tags.

1.4 Issues that you had to overcome

The primary challenge is the mismatch between WSD API output and English source tokens. We use a substitution strategy to replace the words which can cause the error tokenization when using WSD API.

The secondary challenge is API integration and response handling. The AMUSE web sevice requires JSON format files and proper error handling to manage potential network timeouts or malformed responses. We implmented robust request handling with appropriate headers and response check.

1.5 Table results

Metric	**Score**
Overall F1-Score	0.576
Total Instances Evaluated	1,644
Correct Predictions	947
Incorrect Predictions	697
Coverage	100%

表 1: WSD Performance Results on SE13 Dataset

From this paper,we can find that overall score should be 0.576, with 947 correct predictions as well as 697 Incorrect predictions. Moreover, we can find that we have covered all the tokens with gold sense tags. The accuracy 0.576 indicates a moderate performance level.

Comparison with several baseline:

- **Random walk**: Random walk typically achieves 20-30% on polysemous datasets
- **Human performance**: Inter-annotator agreement often ranges from 85-95%

Therefore, it is more accurate than the random algorithm and less accurate than human performance.

1.6 Error analysis

The following is the factors that may cause the the error

Polysemous Word Disambiguation: Polusemous word have more than one sense, so it contributes more to errors.

Preprocessing-Induced Errors: We use the replacement strategy to make the small adjustment of the original English source sentences. Therefore, it may replace the gold annotated tokens into '?' . The character '?' 's synset id is 0.

Domain-Specific Terminology: The source is from news, biographical texts, and web content. The vocabulary is professional from the source.

Examples of typical or interesting errors

- **Polysemous Word Disambiguation:** "plan", "focus", "release", "world"
- **Preprocessing-Induced Errors:** 'greenhouse gases', 'working group', 'global warming'
- **Domain-Specific Terminology:** 'Washington', 'Technology'

1.7 Reflection

strengths

- **It is easy to implement**
- **It costs a little**
- **It has low time complexity**

weaknesses

- **Pre-processing phase may lose some useful tokens**
- **It may ignore some corners cases in pre-processing phase**
- **The accuracy is lower than humane performance**

limits of the method

- **API dependency**: Relies entirely on AMUSE's pre-trained models without possibility for domain-specific fine-tuning
- **Tokenization mismatch**: Fundamental incompatibility between AMUSE's tokenization and source data creates performance ceiling
- **Context limitations**: Cannot capture long-range semantic dependencies needed for complex disambiguation

2 Translation

2.1 Brief Summary

First, we use google translation to translate each english sentence into chinese. Then, we use cometKiwi to evaluate these translation. Then, cometwiki will provide translation scores overall, as well as scores for individual pairs.

2.2 Setup of your evaluation experiments

Step 1: Initialize the Google translator First, we import Translator from googletrans. Then, we configure the translator with the following parameters:

1. Service URL: `translate.google.com`
2. User agent: Standard Chrome browser identifier
3. Timeout: `30` seconds
4. Raise exception: `True`

Choices to use the parameter

- Timeout prevents hanging on slow responses
- Raise exception to automatically throw exceptions on translation failures

Step 2: Pre-processing Data We separate the sentences into several batches. For each sentence in a batch, we perform preprocessing:

Whitespace removal: `text = str(text).strip()` removes leading and trailing whitespace

Space normalization: `text = ' '.join(text.split())` consolidates multiple spaces into single spaces

Step 3: Json format file output

The translation results are saved in JSON format with the following structure:

- `"src"`: Original English sentence
- `"mt"`: Translated Chinese sentence

Step 4: cometKiwi evaluation

We use cometKiwi to evaluate the json format file . The cometKiwi returns the scores for each pair of the sentences.

2.3 Statistics of the data sets and annotations

- Number of sentences 301
- System score 0.75

- Score range 0.172 - 0.893
- High quality sentences >=0.8 128
- Good quality sentences 0.7-0.8 93
- Moderate quality sentences 0.6-0.7 51
- Low quality sentences <0.6 29

2.4 Issues that you had to overcome

- Late response of the server we set the timeout to be 30 seconds
- Error response of the server we use try-except to deal with this issue

2.5 Table Results

表 2: Translation Quality Results

Metric	**Value**
Total Sentences	301
CometKiwi System Score	0.750
Standard Deviation	0.109
Score Range	0.172–0.893
High Quality (≥ 0.8)	128 (42.5%)
Good Quality (0.7–0.8)	93 (30.9%)
Low Quality (< 0.6)	29 (9.6%)
Success Rate (≥ 0.7)	73.4%

2.6 Error Analysis

Analysis of translation failures reveals two main error categories:

Complete Translation Failures

Sentences with CometKiwi scores below 0.2 (e.g., sentences 264, 41) were not translated at all, returning the original English text. These failures occurred due to:

- Sentence complexity and excessive length
- API processing errors

Proper Noun Translation Issues

Sentences scoring 0.4-0.6 contained numerous proper nouns and technical terms that the system struggled to handle appropriately. Common issues included:

- Leaving proper nouns untranslated in Chinese output
- Inconsistent handling of technical terminology
- Uncertainty between transliteration and semantic translation approaches

2.7 Reflection

The overall translation quality achieved a CometKiwi system score of 0.750, indicating a good quality for most of the sentences.

Where Google Translate Worked Best: The translator works best when the sentence is straightforward and easy to understand.

Where Google Translate Worked Worst: The translator works worst when the sentence is with a lot of terminology and difficult to understand.

3 Word Alignment

3.1 Brief Summary

First, we load the source sentences and their Chinese translations into two separate files. Since Chinese text lacks word boundaries, perform word segmentation on the Chinese translations before alignment to enable proper tokenization. Then use SimAlign to align the source sentences with the Chinese translations.

3.2 Setup of the Method

Hyperparameters we tried

1. model: `bert-base-multilingual-cased`

2. distortion: `0`
3. null align: `1`
4. token type: `bpe`
5. matching methods: `a (argmax)`
6. num-test-sents: `None`
7. batch size: `100`
8. log: `false`
9. device: `cpu`
10. add probs: `false`
11. layer: `8`

3.3 Evaluation and Error Analysis

We found some successes and errors in the alignment output, here are some examples:

1. 'U.N. group drafts plan to reduce emissions' and the Chinese translation' 联合国小组草稿计划减少排放', the word 'group' and ' 小组' are aligned correctly, but the word 'to' does not align with any token in the Chinese translation.

2.

Complete Translation Failures

Sentences with CometKiwi scores below 0.2 (e.g., sentences 264, 41) were not translated at all, returning the original English text. These failures occurred due to:

- Sentence complexity and excessive length
- API processing errors

Proper Noun Translation Issues

Sentences scoring 0.4-0.6 contained numerous proper nouns and technical terms that the system struggled to handle appropriately. Common issues included:

- Leaving proper nouns untranslated in Chinese output
- Inconsistent handling of technical terminology
- Uncertainty between transliteration and semantic translation approaches

3.4 Reflection

The overall translation quality achieved a CometKiwi system score of 0.750, indicating a good quality for most of the sentences.

Where Google Translate Worked Best: The translator works best when the sentence is straightforward and easy to understand.

Where Google Translate Worked Worst: The translator works worst when the sentence is with a lot of terminology and difficult to understand.