Bitext Processing: Translation, Alignment, and Word Sense Disambiguation

Anonymous ACL submission

1 Word sense disambiguation

1.1 Method description

Data Preparation We load sentences and tokens from Excel files, formatting as JSON with `'lang': 'EN'`.

AMUSE API Integration Data are sent to the AMUSE WSD API, returning tokenized text with BabelNet synset IDs.

Token Alignment and Output We apply character-level matching with hardcoded corrections for mismatches (e.g., "'s" "s", number formatting), writing synset IDs for tokens with instance IDs to `wsd_output.txt`.

Evaluation We compare output against `se13.key.txt` using `evaluate_wsd.py`.

1.2 Accuracy

The system achieved an accuracy of 0.621.

1.3 Examples of typical or interesting errors

Polysemous Word Disambiguation: Words with multiple senses (e.g., "plan", "world") contribute significantly to errors.

Domain-Specific Terminology: Professional vocabulary from specialized domains (e.g., 'Washington', 'Technology') poses significant disambiguation challenges.

1.4 Reflection

Strengths: Direct API integration with straightforward token matching achieves full coverage.

Weaknesses: Hardcoded corrections are brittle and non-generalizable, resulting in 62.1% accuracy.

Limitations: Manual tokenization fixes cannot scale to new data, and API dependency prevents domain-specific optimization.

2 Translation

Method Description We configure GoogleTranslator with `source='auto'` and `target='zh-CN'`, preprocess sentences by normalizing whitespace, and translate sequentially with error handling. Results are saved to `translations.txt` and evaluated with CometKiwi, producing `translation_scores.txt` with one score per line.

2.1 Evaluation results

The dataset contains 301 sentences with a system score of 0.749 (range: 0.172-0.887, SD: 0.117). Quality distribution: 130 high ($\geq$0.8, 43.2%), 86 good (0.7-0.8, 28.6%), 48 moderate (0.6-0.7, 15.9%), and 37 low ($<$0.6, 12.3%).

2.2 Error Analysis

Two main error categories were identified:

Complete Translation Failures: Sentences 41 and 264 scored below 0.2, returning empty or untranslated text due to complexity and API errors.

Proper Noun Translation Issues: Sentences scoring 0.4-0.6 left proper nouns untranslated or inconsistently transliterated.

2.3 Reflection

Strengths: Achieves 0.749 system score with simple sequential processing and robust error handling.

Weaknesses: Performance degrades on terminology-heavy content (12.3% scored below 0.6), with two complete failures.

Limitations: Lacks domain-specific tuning, relies on generic GoogleTranslator without customization, and sequential processing is slower than batch methods.

3 Word Alignment

3.1 Setup of the Method

Hyperparameters we tried

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

3.2 Evaluation and Error Analysis

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

1. In the 58th sentence, 10 and 10 are aligned correctly, oil and are aligned correctly, and companies and are aligned correctly. 2. In the 13th sentence, with and are incorrectly aligned. 3. In the 58th sentence, "foreign" aligns to and "" at the same time. The alignment with is correct, but the alignment with is incorrect.

As mentioned above, two typical alignment errors are function words and one-to-many mappings. The former is because Chinese lacks direct equivalents for many English function words. The latter is because itermax's purely distributional approach leads to over-alignment when multiple target tokens have similar embeddings to the source. In order to improve alignments, we could add function word filtering by using POS tags to prevent articles and prepositions from aligning to content words. Besides, we could also incorporate compound word detection to treat multi-word English expressions and Chinese character compounds as single alignment units, reducing spurious one-to-many mappings.