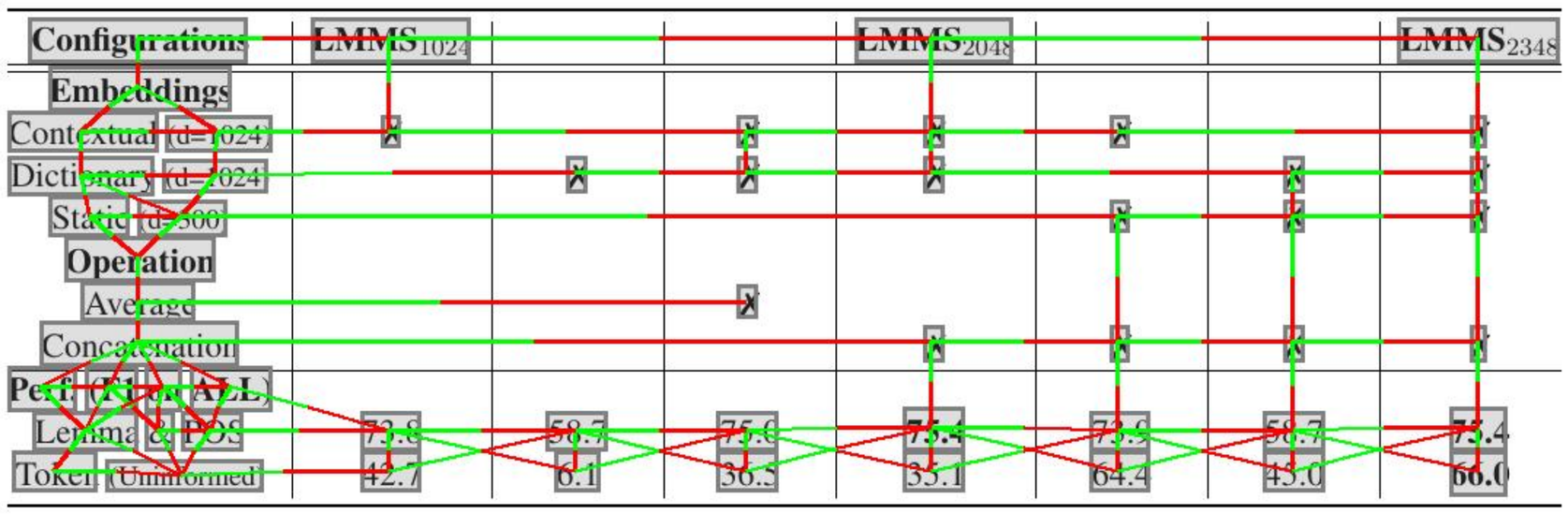

Table 2: Overview of the different performance of various setups regarding choice of embeddings and combination strategy. All results are for the 1-NN approach on the ALL test set of Raganato et al. (2017a). We also show results that ignore the lemma and part-of-speech features of the test sets to show that the inclusion of static embeddings makes the method significantly more robust to real-world scenarios where such gold features may not be available.

4.4 Morphological Robustness

WSD is expected to be performed only against the set of candidate senses that are specific to a target word's lemma. However, as we'll explain in §5.3, there are cases where it's undesirable to restrict the WSD process.

We leverage word embeddings specialized for morphological representations to make our sense embeddings more resilient to the absence of lemma features, achieving increased robustness. This addresses a problem arising from the susceptibility of contextual embeddings to become entirely detached from the morphology of their corresponding tokens, due to interactions with other tokens in the sentence.

We choose fastText (Bojanowski et al., 2017) embeddings (pretrained on CommonCrawl), which are biased towards morphology, and avoid Out-of-Vocabulary issues as explained in §2.1. We use fastText to generate static word embeddings for the lemmas ($\vec{v}_l$) corresponding to all senses, and concatenate these word embeddings to our previous embeddings. When making predictions, we also compute fastText embeddings for tokens, allowing for the same alignment explained in the previous section. This technique effectively makes sense embeddings of morphologically related lemmas more similar. Empirical results (see Table 2) show that introducing these static embeddings is crucial for achieving satisfactory performance when not filtering candidate senses.

Our final, most robust, sense embeddings are thus:

$$\vec{v}_s = \begin{bmatrix} ||\vec{v}_s||_2 \\ ||\vec{v}_d||_2 \\ ||\vec{v}_l||_2 \end{bmatrix}, dim(\vec{v}_s) = 2348$$

5 Experiments

Our experiments centered on evaluating our solution on Raganato et al. (2017a)'s set of cross-domain WSD tasks. In this section we compare our results to the current state-of-the-art, and provide results for our solution when disambiguating against the full set of possible senses in WordNet, revealing shortcomings to be improved.

5.1 All-Words Disambiguation

In Table 3 we show our results for all tasks of Raganato et al. (2017a)'s evaluation framework. We used the framework's scoring scripts to avoid any discrepancies in the scoring methodology. Note that the k-NN referred in Table 3 always refers to the closest neighbor, and relies on MFS fallbacks.

The first noteworthy result we obtained was that simply replicating Peters et al. (2018)'s method for WSD using BERT instead of ELMo, we were able to significantly, and consistently, surpass the performance of all previous works. When using our method (LMMS), performance still improves significantly over the previous impressive results (+1.9 F1 on ALL, +3.4 F1 on SemEval 2013). Interestingly, we found that our method using ELMo embeddings didn't outperform ELMo k-NN with MFS fallback, suggesting that it's necessary to achieve a minimum competence level of embeddings from sense annotations (and glosses) before the inferred sense embeddings become more useful than MFS.

In Figure 2 we show results when considering additional neighbors as valid predictions, together with a random baseline considering that some target words may have less senses than the number of accepted neighbors (always correct).