ATNN Homework 4 – SemEval 2026 Task 5
Rating Sense Plausibility of Words in Ambiguous Narratives

Gavril Ştefan-Dorian

January 2026

Abstract

Task 5 of SemEval-2026 asks participants to submit solutions that predict plausibility ratings (1–5) for candidate word senses in short stories where ambiguity is often real, not just noise. The proposed solution uses DeBERTa-v3-Large, reads the full story context together with a candidate sense description (definition + example) and is trained on the full distribution of annotator ratings using KL-divergence, which fits the task better because it preserves uncertainty.

My best solution achieved Spearman $\rho = 0.607$ and accuracy 73.5% on the hidden test set, trained on a single GPU.

1 Introduction

Word Sense Disambiguation (WSD) is often treated as "pick the one correct sense from a list." In practice (especially in short narratives), the right meaning is not always spelled out: authors leave things implicit, endings can stay open, and multiple interpretations remain plausible. SemEval-2026 Task 5 makes this explicit by asking for *graded plausibility ratings* (1–5) instead of a single label.

Two parts of the task really influence how to approach it:

- **Disagreement is a signal:** we get multiple annotator ratings per instance, so the label is naturally a distribution, not one "truth."
- **Ranking matters:** one of the metrics is Spearman correlation between predictions and averaged human ratings, so it is important to get the ordering right.

The goal was to build a solid, reproducible solution that does not require high computational resources (the proposed solution runs on a single GPU), and to test what actually helps through ablations and robustness analysis (not just over-tuning to dev).

2 Task and Data

Each instance is a short story containing a homonymous target word. The inputs are formed of:

- **Precontext:** a few sentences setting the scene
- **Ambiguous sentence:** sentence with the target word
- **Ending (optional):** can clarify an interpretation or sometimes contradict it

Each candidate sense is given as a **definition** and usually an **example** sentence. Annotators rate how plausible the candidate sense is for the story on a 1–5 scale.

The official split contains 2,280 training, 588 development, and 930 test instances. The metrics used are Spearman rank correlation between predicted scores and averaged human scores, aswell as accuracy within Standard Deviation.

3 Related Work

3.1 Standard Word Sense Disambiguation

Traditional WSD approaches mostly relied on dictionaries or trained separate classifiers for every single word [1, 3]. These methods treat meaning as a multiple-choice question where only one answer is correct. This does not work well for the current task, where a word might be ”somewhat plausible”.

Newer neural models (like BERT) improved this by using context, but they still typically force the model to pick a single winner [5]. The struggle comes from the gray areas found in the AmbiStory dataset, where multiple interpretations can be valid at the same time.

3.2 Using Glosses (Definitions) as Input

A major improvement in this field was GlossBERT [7]. Instead of just guessing a sense ID, GlossBERT feeds the definition (gloss) into the model right next to the sentence. This allows the model to explicitly compare the context with the definition. I use a similar strategy as it lets the model ”read” the definition to judge if it fits, rather than just memorizing sense labels.

3.3 Plausibility and Ranking

Some research argues that word meaning is not binary (True/False) but graded [9]. For example, the benchmark PROBELM [13] tests models on how well they can *rank* likely scenarios. This matches the goal of Task 5: we care more about ranking the senses correctly (Spearman correlation) than getting the exact absolute score perfect. This influenced the decision to look at ranking losses and distributions.

3.4 The AmbiStory Dataset

Unlike simpler datasets that test basic commonsense, AmbiStory requires the model to track the plot across several sentences. Because the stories are tricky, human annotators often disagree. This disagreement should be regarded as useful data (uncertainty) rather than noise to be removed.

4 Modeling Choices

This section explains the reasoning behind the architecture and training decisions.

4.1 Why DeBERTa instead of BERT?

The main reason is how DeBERTa handles position. Standard BERT adds position information directly to the word embeddings, which can sometimes confuse the model about relative distances. DeBERTa keeps content and position separate [14].

In AmbiStory, the ”clue” that resolves the ambiguity is often in the final sentence (the ending), far away from the target word. DeBERTa’s attention mechanism is generally better at connecting these distant cues than previous models.

4.2 Cross-Encoder vs. Bi-Encoder

The Story and the Definition are fed into the model at the same time: `[CLS] Story [SEP] Definition [SEP]`

The alternative (Bi-Encoder) processes them separately and just compares their vectors at the end. While Bi-Encoders are faster, Cross-Encoders are much more accurate because the model’s self-attention can compare every word in the story to every word in the definition. Since the dataset is small, we can afford the extra computational cost of the Cross-Encoder to get better accuracy.

4.3 Pooling Strategy

The `[CLS]` token output is used to represent the entire pair. This is the standard way to do classification in Transformers. Since the self-attention mechanism allows the `[CLS]` token to aggregate information from both the story and the definition, it contains enough information to predict the final score.

4.4 Why Train on Distributions (KL-Divergence)?

The most important design choice was to train on the distribution of human ratings instead of just the average score.

If 5 annotators vote `[1, 1, 5, 5, 5]`, the average is 3.4. If we train the model to predict 3.4, it learns that the word is "neutral." In reality, the word is highly controversial.

By using KL-Divergence, the model predicts the histogram of votes. This teaches the model to be uncertain when humans are uncertain. If the story is clear, the model learns to be certain of one vote. If the story is vague, the model spreads its probability out. This leads to a more robust model than simple regression.

5 Method

5.1 Cross-encoder model

I used DeBERTa-v3-Large as a cross-encoder by concatenating the story parts and pairing them with the candidate sense text:

```
[CLS] Context: <story> [SEP]
Meaning: <definition> Example: <example> [SEP]
```

The `[CLS]` representation is then sent through a linear layer to produce 5 logits (ratings 1–5).

5.2 Distributional supervision with KL-divergence

Compute the empirical distribution of annotator ratings, instead of training on a single averaged score:

$$p_{\text{gold}}(k) = \frac{\#\{\text{votes for } k\}}{n}, \quad k \in \{1, 2, 3, 4, 5\}.$$

Let p_θ be the model prediction (softmax over logits). The following are then minimized:

$$\mathcal{L}_{\text{KL}} = \text{KL}(p_{\text{gold}} \| p_\theta) = \sum_{k=1}^{5} p_{\text{gold}}(k) \log \frac{p_{\text{gold}}(k)}{p_\theta(k)}.$$

For a scalar prediction (used to compute correlation and also for submission), I compute the expected rating:

$$\hat{s} = \sum_{k=1}^{5} k \cdot p_\theta(k),$$

and round to the nearest integer when generating the official prediction file.

5.3 Ranking loss (exploratory)

Since Spearman is rank-based, I also experimented with a pairwise margin ranking loss within groups of story variants (same `sample_id`). In practice, this only helps if batches contain multiple variants from the same group, so it requires special batching; otherwise the ranking signal is sparse and noisy. As such, the setup that made ranking effective improved dev scores but overfit and reduced test performance (Section 8.2).

6 Experimental Setup

Training used AdamW, cosine learning-rate schedule with warmup, mixed precision, and gradient accumulation to reach an effective batch size of 16.

- Base model: `microsoft/deberta-v3-large`
- Max length: 256
- Batch size: 8 with grad accumulation 2 (effective 16)
- Learning rate: 2×10^{-5} (cosine decay), warmup 6%
- Epochs: 3 (final model), up to 6 for one ablation run
- Loss: KL-divergence (final); KL + ranking (ablation)

7 Results

7.1 Official hidden test performance

The best official submission achieves:

$$\rho = 0.607, \qquad \text{Accuracy} = 73.5\%.$$

	Spearman ρ	Accuracy
DeBERTa cross-encoder + KL (final)	0.607	0.735

Table 1: Official hidden test-set result.

7.2 Training progression (final run)

Epoch	Train Loss	Dev ρ	Dev Acc
1	0.3838	0.4732	0.6531
2	0.3376	0.6055	0.7023
3	0.2733	**0.6234**	0.7142

Table 2: Training progression for the final KL-supervised model.

7.3 Official baseline comparison

Model	Spearman ρ	Acc. w/in SD
Random	0.000	0.454
Majority	–	0.558
LLaMA-3 (0-shot)	0.462	0.663
Mistral (0-shot)	0.382	0.568
Mixtral (0-shot)	0.606	0.634
GPT-4o-mini (0-shot)	0.726	0.726
GPT-4o (0-shot)	0.756	0.755
o3 (0-shot)	0.753	0.763
DeepSeek (0-shot)	0.740	0.790
LLaMA-3 (4-shot)	0.491	0.694
Mistral (4-shot)	0.209	0.522
Mixtral (4-shot)	0.607	0.649
GPT-4o-mini (4-shot)	0.737	0.726
GPT-4o (4-shot)	0.742	0.725
o3 (4-shot)	0.742	0.760
DeepSeek (4-shot)	**0.767**	**0.816**
DeBERTa cross-encoder (ours)	**0.607**	**0.735**
Human upper bound	0.834	0.892

Table 3: Benchmark comparison with baselines and published AmbiStory results. Large language model scores are reported from the AmbiStory paper for reference.

8 Ablation Study and Analysis

8.1 What influenced the best result (and why)

This section directly answers the presentation question: what mattered most and why.

1) Training on distributions (KL) helped. Averaging labels encourages the model to be confident about that average. KL-divergence however, keeps the "spread" of ratings, so the model can learn that some cases are genuinely uncertain. This seems especially important in examples with high disagreements.

2) Full story context helped. A lot of stories only make sense once you read the precontext or the ending. If you remove those, the model is forced to guess based mostly on the ambiguous sentence.

3) Shuffling beat grouped batching. Grouped batching makes ranking loss feasible, but it also removes randomness and makes it easier for the model to memorize patterns on the dev set. In local runs it looked better on dev but generalized worse to the hidden test set.

8.2 Training configuration ablation

Here two training configurations are compared to understand the trade-off between enabling ranking supervision and generalization, even with more training time offered for the former:

Configuration	Epochs	Dev ρ	Test ρ	Test Acc	Time (s)
GroupBatchSampler + Ranking	6	0.665	0.589	0.703	1341
Standard Shuffling (final)	3	0.623	**0.607**	**0.735**	622

Table 4: Key ablation over training configurations. Grouped batching enables ranking loss but overfits; standard shuffling generalizes better.

Grouped batching improved dev Spearman by +0.042, but dropped on test by -0.018, which looks like dev overfitting. Standard shuffling had slightly lower dev scores but better hidden test performance.

8.3 Robustness slices (dev set)

Robustness slices were created on the dev set to see where the model struggles most.

Slice	Spearman ρ	N
Ended stories (ending present)	0.669	310
Open-ended (no ending)	0.619	278
Low disagreement ($\sigma <$ median)	0.729	294
High disagreement ($\sigma \geq$ median)	0.513	294
Short inputs (Q1)	0.653	147
Long inputs (Q4)	0.653	147

Table 5: Robustness slices on the dev set. The biggest drop is on high-disagreement items.

9 Qualitative Error Analysis

These examples are not meant to be exhaustive, but they match what is seen in the robustness slices and in common error patterns.

9.1 Example 1

Story excerpt: "Lily was sitting by the pond... A frog appeared on a rock... **The frog began to croak.**"

Candidate sense: "pass from physical life and lose all bodily attributes"

Human mean: 1.0 *Model prediction*: 4

Analysis: The story strongly suggests the literal animal sound (pond, frog, wildlife), but the model still assigns high plausibility to the figurative "die" meaning. As such, the model sometimes does not integrate obvious narrative evidence as strongly as it should.

9.2 Example 2

Story excerpt: "He had waited a long time to have this rare beef. **It was cooked to medium [..]**."

Candidate sense: "(of meat) cooked a short time; still red inside"

Human mean: 2.0 *Model prediction*: 5

Analysis: "Cooked to medium" contradicts the "rare" cooking sense, but the model predicts maximum plausibility anyway. This is a good example of keyword fixation (seeing "rare beef") plus weak handling of contradictory endings.

9.3 Example 3

Story excerpt: "...he leaned over the table and **snorted loudly**... Mike tried to hide the small bag."

Candidate sense: "inhale recreational drugs"

Human mean: 2.6 ($\sigma = 1.82$) *Model prediction*: 5

Analysis: Humans disagree a lot here (snorted could be embarrassment or drugs), but the model still predicts extreme certainty. This matches the biggest weakness from Table 5: performance drops heavily on high-disagreement items.

10 Computational Efficiency

Metric	GroupBatchSampler	Standard Shuffling
Training time (total)	1341 s (∼22.4 min)	622 s (∼10.4 min)
Epochs used	6	3
Peak GPU memory	∼11.2 GB	∼11.2 GB
GPU utilization	near-saturated	near-saturated

Table 6: Efficiency comparison on a single Tesla T4 GPU (from logs).

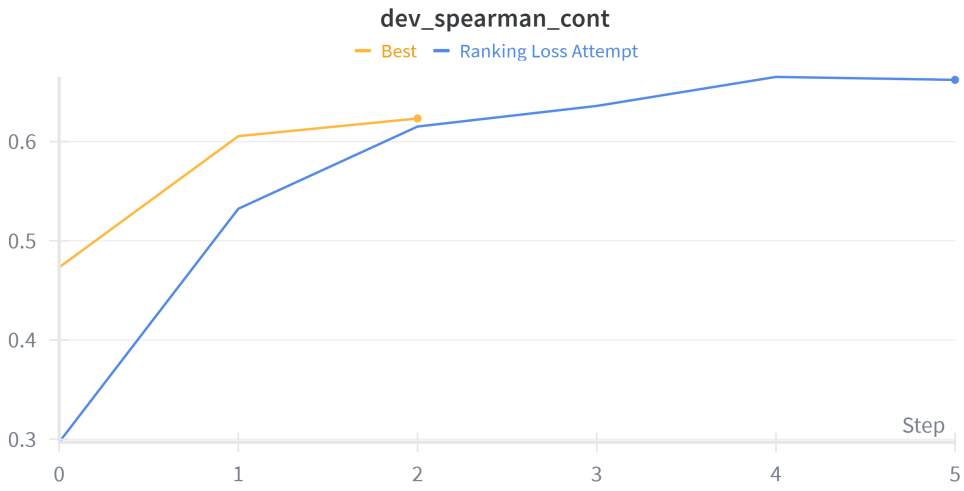

Figure 1: Dev Spearman correlation across epochs.

Figure 2: Training loss across epochs.

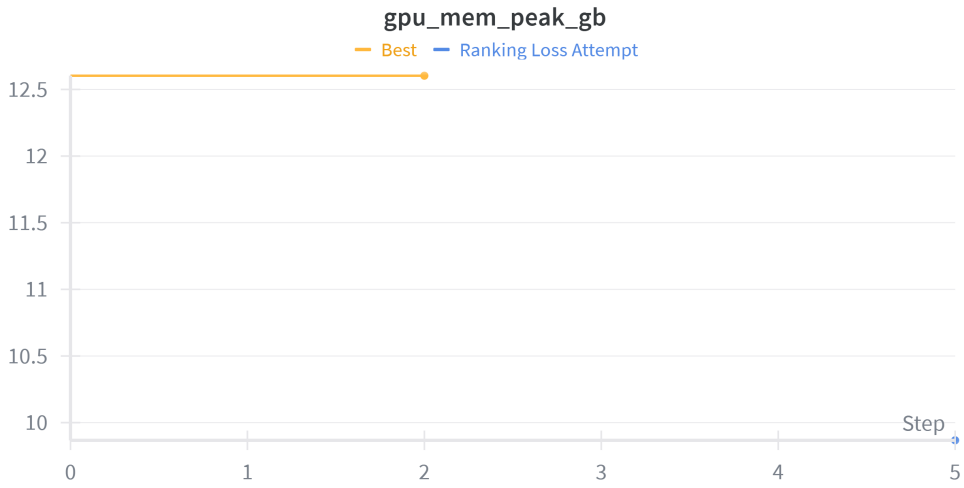

Figure 3: Peak GPU memory usage during training.

11 Discussion and Future Work

The main limitation is dealing with truly ambiguous stories where humans disagree heavily. With more time, the most promising directions would be:

- **Uncertainty-aware training:** encourage broader output distributions when evidence is weak, instead of pushing toward extreme ratings.
- **Story-aware batching without losing shuffle:** keep randomization but still group variants inside shuffled batches so ranking loss can work better.
- **Contradiction-aware training:** add explicit signals for endings that contradict a candidate sense, so the model learns to downweight those cases.

12 Conclusion

The proposed solution is a DeBERTa-v3-Large cross-encoder for graded word-sense plausibility rating, training on full annotator distributions using KL-divergence. The best submission reaches Spearman $\rho = 0.607$ and accuracy 73.5% on the hidden test set with single-GPU training. Ablations show that distributional supervision and full context matter most, while ranking-based losses require careful batching and can easily overfit. The main challenge is modeling uncertainty in high-disagreement examples.

References

[1] Michael Lesk. Automatic sense disambiguation using machine readable dictionaries. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, pages 24–26, 1986.

[2] Roberto Navigli. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69, 2009.

[3] Zhi Zhong and Hwee Tou Ng. It makes sense: A wide-coverage word sense disambiguation system for free text. In *Proceedings of the ACL 2010 System Demonstrations*, pages 78–83, 2010.

[4] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL-HLT 2018*, pages 2227–2237, 2018.

[5] Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. Neural sequence learning models for word sense disambiguation. In *Proceedings of EMNLP 2017*, pages 1156–1167, 2017.

[6] Mikael Kågebäck and Hans Salomonsson. Word sense disambiguation using a bidirectional LSTM. *arXiv preprint arXiv:1606.03568*, 2016.

[7] Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of EMNLP-IJCNLP 2019*, pages 3509–3514, 2019.

[8] Fuli Luo, Tianyu Liu, Zexue He, Qiaolin Xia, Zhifang Sui, and Baobao Chang. Leveraging gloss knowledge in neural word sense disambiguation by hierarchical co-attention. In *Proceedings of EMNLP 2018*, pages 1402–1411, 2018.

[9] Katrin Erk, Diana McCarthy, and Nicholas Gaylord. Investigations on word senses and word usages. In *Proceedings of ACL-IJCNLP 2009*, pages 10–18, 2009.

[10] David Jurgens and Ioannis Klapaftis. SemEval-2013 Task 13: Word sense induction for graded and non-graded senses. In *Proceedings of SemEval 2013*, pages 290–299, 2013.

[11] Janosch Gehring and Michael Roth. AmbiStory: A challenging dataset of lexically ambiguous short stories. In *Proceedings of *SEM 2025*, pages 152–171, 2025.

[12] Melissa Roemmele, Cosmin Adrian Bejan, and Andrew S. Gordon. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium Series*, 2011.

[13] Zhangdie Yuan, Eric Chamoun, Rami Aly, Chenxi Whitehouse, and Andreas Vlachos. PROBELM: Plausibility ranking evaluation for language models. In *Proceedings of COLM 2024*, 2024.

[14] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. DeBERTa: Decoding-enhanced BERT with disentangled attention. In *Proceedings of ICLR 2021*, 2021.