Sabanci_group4 at SemEval-2026 Task 5: Uncertainty-Aware Semantic Plausibility Scoring via GNLL Regression and LLM Rationales

Salih Numan Büyükbaş[†], **Doruk Benli**[†], **Osman Kara, Dilara Keküllüoğlu**
Faculty of Engineering and Natural Sciences, Sabancı University, Türkiye
{salihnuman, dorukbenli, osmankara, dilara.kekulluoglu}@sabanciuniv.edu

Abstract

SemEval-2026 Task 5 is a shared task on rating the plausibility of an ambiguous homonym in a predetermined context. The dataset of this task consists of a precontext & sentence & ending combinations for each homonym, and the plausibility of the sample is manually rated by 5 annotators. The task of participating teams was to automatically predict the plausibility with respect to the mean rate given by the annotators. Unlike traditional models that rely on single-label selection, this task frames disambiguation as a probabilistic distribution over multiple plausible meanings. To this end, we propose an uncertainty-aware training strategy using GNLL regression, and semantic context enrichment through POS tags and LLM rationales. Our system exhibits competitive performance, achieving 90% accuracy within standard deviation and 81% Spearman correlation, and placing us in the fifth place in the leaderboard.[1]

1 Introduction

SemEval-2026 Task 5 challenges participants to rate the plausibility of specific word senses within the ambiguous English contexts provided by the AmbiStory dataset (Gehring et al., 2026). Each data point consists of a target homonym, a precontext of 3 sentences, an ambiguous sentence that the homonym is used in, an optional ending, and a judged meaning. The task is to predict the plausibility of the given judged meaning with respect to the context, where the ground-truth is derived from the aggregated judgments of five human annotators per sample. Addressing this task is important, as natural language often supports multiple plausible interpretations. However, current language models are optimized to choose single 'correct' sense, disregarding the underlying distribution of multiple plausible meanings.

In this paper, we model the task as a regression problem targeting the mean human plausibility score. We explore different modeling approaches that integrate predictive uncertainty, contrasting their performance against baseline models that are optimized solely for point estimates of the mean. We employ Gaussian Negative Log-Likelihood (GNLL) loss, which enables our system to simultaneously model both the mean plausibility and the uncertainty inherent in the human annotations. Furthermore, we explore the teacher-rationale augmentation techniques by enriching each instance with synthetic explanations generated by a Large Language Model (LLM) to provide additional semantic context. We also utilize the augmentation of Part-of-Speech (POS) tags to provide grammatical context and increase the consistency of model predictions.

Our main discovery is that in a prediction task where an inherent uncertainty and subjectivity is present, formulating the task as a regression often leads to suboptimal performance. Instead,we argue for a reformulation that guides the model to explicitly learn and predict the degree of ambiguity within each sample. Our method achieves an accuracy within standard deviation ($Acc@std$) of 90% and a Spearman correlation of 81%, placing us in the fifth place in the leaderboard. Qualitative analysis reveals that in nearly half of the instances, our model's predictions fall within a margin of ± 0.4 of the human mean. Furthermore, error cases are primarily concentrated in 'perfect consensus' scenarios—where all five annotators assigned a score of 1 or 5—indicating that the model's uncertainty-aware calibration occasionally struggles with these extreme, unambiguous cases.

[†]Both authors contributed equally to this work; the order of the first two authors was decided by a coin flip.

[1]Our code is publicly available at https://github.com/DorukBenli/semeval26-05-scripts

Proceedings of the The 20th International Workshop on Semantic Evaluation (2026), pages 1056–1062

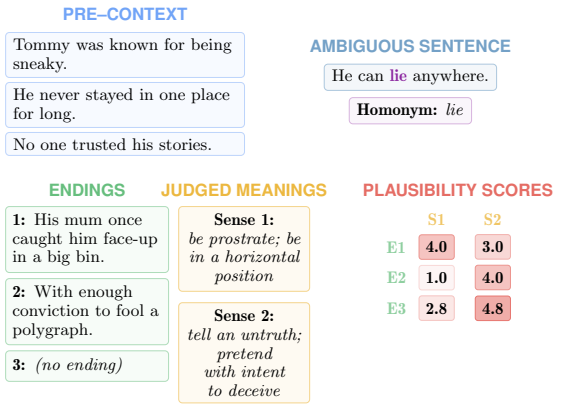

Figure 1: An example from the AmbiStory dataset.

2 Background

Word Sense Disambiguation has traditionally been framed as selecting a single correct sense for an ambiguous word given its context (Navigli, 2009). The AmbiStory task departs from this formulation by requiring systems to predict graded plausibility scores rather than discrete sense labels. Recently, pretrained language models have achieved strong results on WSD benchmarks by framing it as a sequence classification or nearest-neighbor problem (Blevins and Zettlemoyer, 2020). However, these approaches still assume a single gold sense per instance and do not model the degree of plausibility across multiple interpretations, which is the core challenge of the AmbiStory task.

The AmbiStory dataset consists of short narratives that deliberately include lexical ambiguity. Each story begins with a three–sentence pre–context, followed by a sentence containing a homonym. Depending on the surrounding context, various interpretations may feel reasonable. Stories also include an optional ending sentence, which can push the reader toward a specific meaning. Each core story can appear with two different endings or with no ending. For every story setup, annotators rated how plausible each meaning was in the given context and ending on a scale from 1 to 5. Each meaning-–story pair received at least five human ratings. Because each pre–context can combine with two meanings and three possible ending conditions, the dataset includes six annotated samples per story. Visualization of one sample story with six different combinations is presented in Figure 1. Our training and development datasets include 2280 and 588 instances in total, respectively. The test set, released just before the evaluation phase, contains 930 instances.

3 System Overview

Our proposed system architecture consists of three primary components designed to capture the lexical ambiguity: GNLL Loss formulation that incorporates the uncertainty modeling, POS taggings that enrich the input with a key information for the sense disambiguation, and the LLM–generated rationales that expand the contextual information available for each instance.

3.1 GNLL Loss

The task in its essence requires the system to be able to model the uncertainty of the given instance, by integrating not only the mean of annotator predictions, but also the standard deviation of them. Defining the problem as a prediction of just one value in $[0, 5]$ does not fully reflect the complexity of the task. The model should also learn the uncertainty of its own predictions for each instance, and indeed should be enforced to minimize its uncertainty so as to actually be learning the task. The intuition of our design is that we aim to teach the model to estimate its own predictions' margin of error (to account for instances where multiple senses of a word remain simultaneously plausible in the given precontext & ending), and also teach it to make as certain predictions as possible (to model the regression task). To this end, the work of Kendall and Gal (2017) studied the uncertainty modeling topic and came up with a framework designed for computer vision classification tasks that accurately captures uncertainty. Their proposed *Gaussian Negative Log-Likelihood* loss can also be used for a regression task such as ours. To incorporate this method to our task, we get two outputs from the model: a prediction (mean) and how certain it is (log-variance). Whenever the model predicts with a high variance (uncertain), the penalty is lower, but the loss also regularizes the variance output in order to prevent the model from cheating by outputting large variance for every sample. The loss function is as follows:

$$\mathcal{L}_{\text{GNLL}} = \frac{1}{2\sigma^2}(y - \mu)^2 + \frac{1}{2}\log \sigma^2 \qquad (1)$$

where μ denotes the predicted score for the sample, $\log \sigma^2$ denotes the log-variance output of the model, and y is the correct score, i.e the correct mean of annotator predictions. The first term in the loss enforces the model to predict closer to the correct score, controlled by the variance (i.e uncertainty)

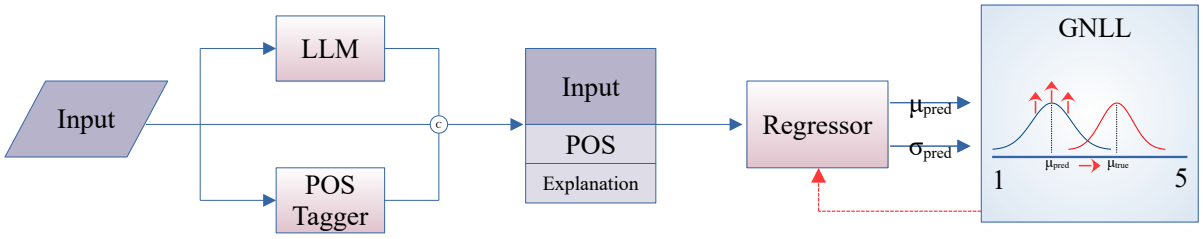

Figure 2: The visualization of the algorithmic flow of our method. The Input samples are first passed through two models (POS tagger and a large LM) to augment the POS tags and teacher explanations. Then, a language model for regression is used with the proposed loss function. The red lines inside the GNLL blocks show the two objective, one being pushing the predicted mean towards the true mean, and the other being minimizing the standard deviation. The Gaussian curves represent the true and predicted distributions, defined over the mean and std values of each. The red dashed-arrow shows the back-propagation of the loss.

of the model. The second term prevents the model to maximize the log-variance instead of finding a pareto-optimal solution by penalizing the high log-variance predictions. Thus, the model can only minimize the loss by predicting either a correct enough score or a wide enough variance.

3.2 POS Taggings

The semantic meaning of a word highly correlates with how and why the word is used in the sentence. Knowing the reason and purpose of using a specific word gives valuable information on the meaning of it. Thus, the Part-of-Speech tag of a word possesses a significant promise for improvement in our task. To this end, we pass our samples through a POS-tagger transformer before training, and in our prompts on the training phase, we also provide the tag given to the word in the sentence. This additional information about the target sentence helps the model understand the consistency requirement of the inferred tag in the meaning of the word with the actual tag of the word in the sentence its used in. Since most POS taggers exceed 95% accuracy, we expect false positives to have a negligible effect.

3.3 Large Model Explanations

Using large models with an API call for the task provided worse results than our methods in predicting the plausibility of the sentences. This shows the effectiveness of our fine-tuning strategy, but we still believe that they can be used as a “teacher model” to guide our smaller model to “what to attend to” or “what reasoning can be made” in order to come to a conclusion. LLM’s are usually "generalists" that fail to provide top-tier performance on the specific tasks compared to smaller models that are trained for the task, but they can enrich the content of the instance and possibly increase the information carried on the features. To harness this, we adopt a pipeline inspired by Wang et al. (2022), who demonstrate that augmenting prompts with step-by-step rationales significantly improves performance and interpretability over standard input-output mappings. By treating a large LLM as a ’teacher’ to produce these reasoning steps, we can enrich our instances with features that guide our smaller model toward the correct disambiguation logic.

3.4 Algorithmic Flow

The workflow of our proposed method is illustrated in Figure 2. During the training phase, we optimize only the regressor’s parameters using Low-Rank Adaptation (LoRA). By freezing the backbone network’s weights and updating only the LoRA adapters, we significantly reduce the computational overhead. Detailed hyperparameters for the LoRA configuration are provided in Appendix A.

To append the POS taggings, we use spacy’s POS tagger transformer (Honnibal et al., 2020). To generate the LLM explanations, we use DeepSeek R3.2’s API (Liu et al., 2024). The prompt we used to get the explanations can be seen in Appendix A.

4 Experiments and Results

In this section, we present the experimental setup and results of our system. We describe the parameters used for the training and different configurations we have used to test our approach. We ran 3 ablations with our final method and reported a comparison. Finally, we conduct an error analysis on the test set to better understand the performance.

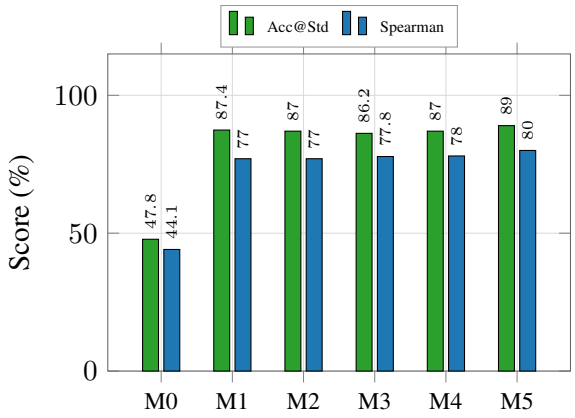

Figure 3: Development set performance comparison across GNLL-based model variants (Accuracy@Std and Spearman). **Model mapping:** M0: Qwen-3-8B (Baseline, zero-shot), M1: Qwen-3-8B (POS + GNLL), M2: Qwen-3-8B (POS + GNLL + Synim), M3: Qwen-3-8B (POS + GNLL + Simyn), M4: Qwen-3-8B (Exp + GNLL), M5: Qwen-3-8B (Exp + POS + GNLL).

4.1 Experiments

The dataset was crafted and already split by the task owners, so no further split was conducted. Across both the development and evaluation phases, our core methodology remained consistent, employing the GNLL loss formulation augmented with POS tags and teacher explanations. The models were fine-tuned for 5 epochs using a batch size of 4 and a learning rate of 5×10^{-5}. For efficient training, we applied LoRA adapters with a rank (r) of 32 and an α of 64.

The only variation between the two phases was the composition of the training data. During the development phase, the model was trained exclusively on the training set and evaluated on the development set. For the evaluation phase, we concatenated the training and development sets to train the model, which was then evaluated on the hidden test set. The evaluation metrics are Spearman correlation and $Acc@std$. The final official evaluation score is the average of the two. All experiments were conducted on an NVIDIA A100 Tensor Core GPU (40GB) via Google Colab, utilizing the Hugging Face transformers and peft libraries. The list of hyperparameters is provided in Appendix A.

4.2 Results

Our final system achieved an $Acc@std$ of 90% and a Spearman correlation of 81% on the hidden test set while achieving 89% $Acc@std$ and 80% Spearman correlation on the development set. With a combined average test score of 85%, our approach demonstrates highly competitive performance despite utilizing a relatively small backbone model. This result secured fifth place on the leaderboard out of 175 participants, trailing the top-ranked system by only 4 percentage points.

Before establishing our final configuration, we conducted extensive experiments across different architectures, maintaining a consistent train/development data split (already given by task owners) throughout trials. Among the candidate backbone models, we observed that the Qwen3-8B architecture consistently yielded the best performance. Furthermore, we conducted ablation studies to quantify the impact of each proposed component within our pipeline. The results of these ablations are presented in Figure 3.

4.2.1 GNLL loss without any semantic augmentation

We established our baseline by fine-tuning the Qwen3-8B model using LoRA adapters with a rank (r) of 32 and an α of 64. This configuration achieved 87.0% $Acc@std$ and a 75.9% Spearman correlation. However, because the GNLL loss framework explicitly models predictive uncertainty, we hypothesized that enriching the input with semantic augmentations would provide the contextual information to improve both metrics.

4.2.2 GNLL Loss with Part-of-Speech (POS) Tags (without LLM rationales)

Building upon the GNLL baseline, we enriched the input by appending the Part-of-Speech (POS) tag of the target word within the ambiguous sentence. Maintaining the Qwen3-8B architecture and LoRA configuration ($r = 32$, $\alpha = 64$), this augmentation increased $Acc@std$ to 87.4% and boosted the Spearman correlation to 77.0%. These results demonstrate that providing explicit grammatical context aids the model in disambiguating the correct semantic sense, especially improving the relative ranking capability.

4.2.3 Unsuccessful Approaches

We have created a set of synthetic data similar to given training data from various LLMs and trained our system by combining training and synthetic data. However, the results did not improve the $Acc@std$. We also explored reinforcing the model towards outputting similar embeddings for the meaning of the word and the sentence the word is used in, so that the sense is better represented in

Table 1: Distribution of prediction distances from ground truth for failed test samples (outside $\pm\sigma$ range).

Distance	Percentage (%)	Cumulative (%)
0.0–0.1	9.31	9.31
0.1–0.2	10.12	19.43
0.2–0.3	15.38	34.82
0.3–0.4	11.74	46.56
0.4–0.5	12.96	59.51
0.5–0.75	19.84	79.35
0.75–1.0	9.72	89.07
1.0–1.5	7.69	96.76
1.5–2.0	2.83	99.60
2.0+	0.40	100.00

the feature space but this also did not improve the $Acc@std$. Results of these cases can be seen from Figure 3 with labels M2 and M3.

4.2.4 Error analysis

Upon the release of the test set, we checked the cases that we were not able to predict within the $Acc@std$ range. We observed that 38% of these failures occurred in cases of perfect human consensus (ratings of 1 or 5 with zero standard deviation). When we checked how far off we were from the actual results, we observed that 89% of the cases were incorrect within a 1 point range, meaning that even if the model failed to get the result correctly, it still predicted a plausibility score that is close to the actual range. For instance, in test case ID 94—which has a human mean of 1.2 and an $Acc@std$ lower bound of 0.75—our model predicted a 0.4. While strictly 'incorrect' by the metric, this output correctly identifies the sense as highly implausible. Consequently, we conclude that our system exhibits strong generalization capabilities across varied levels of semantic ambiguity.

We perform a fine-grained analysis of the test set outputs by creating a binned heatmap (Figure 4), where the diagonal represents instances predicted within the $Acc@std$ range. The upper triangle indicates underestimation of plausibility, while the lower triangle denotes overestimation. The high density on the diagonal suggests that the regressor, augmented with POS tags and LLM rationales, distinguishes plausibility across most narrative contexts. However, the model's performance at the extrema confirms our previous observation on the limitations on consensus cases: it misses 75 instances at each end of the scale, yielding conservative predictions in the ranges $[2, 3]$ and $[3, 4]$ for ground-truth scores in $[1, 2]$ and $[4, 5]$, respectively. Furthermore, a higher concentration of instances in the lower triangle indicates an inclination toward overestimation, potentially caused by the supportive nature of the teacher-generated explanations.

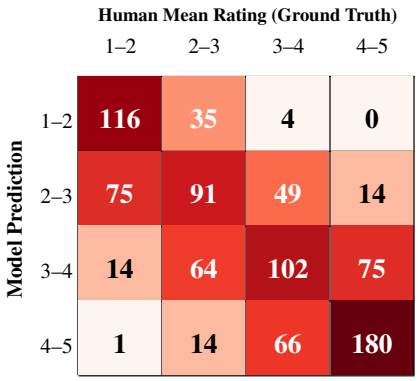

Figure 4: Binned plausibility prediction heatmap on test set

For a qualitative analysis of the proposed method, we present two instances with their predicted and correct plausibility scores, the results of which can be seen in Table 3. Our method shows a significant superiority compared to the baseline, providing close results to the average in low-std cases. In these examples, the ending is not provided, making the instance more difficult to disambiguate compared to the other samples.

5 Conclusion

We presented a regression-based approach for the AmbiStory task using Gaussian Negative Log-Likelihood (GNLL) loss to jointly model plausibility scores and annotator uncertainty. Our best configuration was Qwen3-8B fine-tuned with LoRA (rank=32, α=64) which includes POS tags and teacher-generated explanation cues as semantic augmentations. The final system achieved 90% $Acc@std$ and 81% Spearman correlation on the test set. Results indicate that explicit uncertainty modeling and lightweight semantic enrichment improve performance over standard regression setups. A notable limitation of our model is in the annotator-consensus cases. Because the model optimizes for a distribution, achieving the absolute boundaries of the scale remains a significant challenge. Future research could explore boundary-sensitive loss functions that explicitly penalize deviations at the extrema, encouraging the model to output more confident predictions when ambiguity is low.

References

Terra Blevins and Luke Zettlemoyer. 2020. Moving down the long tail of word sense disambiguation with gloss informed bi-encoders. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1006–1017.

Janosch Gehring, Selina Meyer, and Michael Roth. 2026. SemEval-2026 task 5: Rating plausibility of word senses in ambiguous stories through narrative understanding. In *Proceedings of the 20th International Workshop on Semantic Evaluation*, San Diego, California. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. *spaCy: Industrial-strength Natural Language Processing in Python*.

Alex Kendall and Yarin Gal. 2017. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

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

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, and Denny Zhou. 2022. Rationale-augmented ensembles in language models. *Preprint*, arXiv:2207.00747.

Appendices

A Hyperparameters

The hyperparameters used in the training process of our method are listed in Table 2. The usage of same parameters with the same seed value will lead to the reproduction of the results we report. Most of the parameters' default values are used, while some of them were tuned to get the most consistent performance. Using epochs of 5 gave the most consistent results troughout our experiments, so we sticked to 5 epochs. Furthermore, we provide the specific prompt templates utilized for each model that requires a prompting interface (namely the teacher LLM and the Regressor) to ensure reproducibility. These prompts can be found in Figure 5.

B Other Trials

For the uncertainty modeling, we have experimented with various models as well as the GNLL loss. Mainly, we have builded up on 3 different approaches:

1. **Interval Loss**: A straight-forward implementation that integrates a direct MSE loss, but not penalizing the score if it is inside the mean $\pm$ std range.

2. **Hybrid Loss**: To enforce alignment with annotator predictions, we add a second term to the Interval Loss that calculates the correlation r between the predictions and correct values in the batch, and returns $1 - r$.

3. **KL Divergence Loss**: We also experimented on the performance of directly modeling the annotator scores. By getting 5 outputs from our model and minimizing the KL divergence

Table 2: Hyperparameters and configuration details for the GNLL training setup.

Parameter	**Value**
Model Architecture	
Base Model	Qwen3-8B
Precision	bfloat16
PEFT (LoRA) Configuration	
Rank (r)	32
Alpha (α)	64
Dropout	0.05
Target Modules	q, k, v, o, g, u, d
Bias	None
Optimization & Training	
Optimizer	AdamW
Learning Rate	5×10^{-5}
Scheduler	Linear Decay
Warmup Steps	50
Num Epochs	5
Per-Device Batch Size	4
Gradient Accumulation	2
Effective Batch Size	**8**
Loss Function (GNLL)	
Log-Variance Clamp (Min)	-10.0
Log-Variance Clamp (Max)	5.0
Data & Environment	
Max Sequence Length	512 tokens
Padding Strategy	max_length
Random Seed	42
GPU	NVIDIA A100-SXM4-40GB

Table 3: Qualitative analysis of instances with the proposed GNLL method. Baseline's zero-shot performance is added for comparison.

Instance Details	Evaluation Scores
Example 1 (ID: 89) **Target Word:** *star* **Meaning:** An actor who plays a principal role **Context:** The night sky was full of bright lights. Many gathered in the field with their telescopes and cameras ready. Everyone was hoping for a glimpse of something spectacular. It was clear which one was the outstanding star. **Ending:**	**Gold:** 1.20 (±0.45) **Base:** 5.00 **Ours:** 1.20
Example 2 (ID: 251) **Target Word:** *fiddling* **Meaning:** Play the violin or fiddle **Context:** The meeting started promptly at nine. Everyone in the room was expected to contribute. Yet, some were clearly distracted. Can they stop fiddling? It's really irritating and annoying. **Ending:**	**Gold:** 1.40 (±0.55) **Base:** 5.00 **Ours:** 1.50
Example 3 (ID: 80) **Target Word:** *stars* **Meaning:** (astronomy) a celestial body of hot gases that radiates energy derived from thermonuclear reactions in the interior **Context:** Tom had spent months planning his special trip. He packed his bags, making sure to include his telescope and binoculars. As the day approached, his excitement grew with every passing hour. He looked forward to seeing the stars. **Ending:** He couldn't wait to look up and see all the lights glimmering.	**Gold:** 5.00 (± 0.00) **Base:** 1.00 **Ours:** 4.47

Table 4: Experiment results on dev set with other methods for uncertainty modeling.

Method	$Acc@std$	Spearman
Interval Loss	83.30	73.00
Hybrid Loss	84.60	75.20
KL Div. Loss	87.24	78.30

between each instance's 5 scores and predicted 5 scores, we aimed to model annotators's bias and uncertainty.

The results obtained from using these methods under the same comparison environment is listed in Table 4.

System Prompt Configurations

1. LLM Explanation Generator (Teacher)

System Prompt: You analyze word-sense ambiguity and produce short, dataset-safe explanations. Do NOT reveal hidden chain-of-thought. Provide concise evidence-based steps referencing the given text only. Return ONLY valid JSON.

2. Regressor (Student)

System Prompt: You are a semantic consistency evaluator. Your task is to determine if a story ending is plausible specifically when an ambiguous word in the story is forced to have a provided definition. Output a plausibility score from 1.0 (Impossible with this meaning) to 5.0 (Perfectly logical).

3. Regressor Input Template

`Word: "{homonym}" | POS: {pos} | Definition: {judged_meaning}`
`(Example usage: "{example_sentence}")`

STORY CONTEXT
`Pre-context: {precontext} | Ambiguous Sentence: {sentence}` **### PROPOSED ENDING**
{ending}

INSTRUCTION
Given the definition and part of speech above, how plausible is this ending for the story? Consider if the ending matches the specific meaning of "{homonym}" defined above.

Figure 5: Overview of the prompting strategy, including system instructions for the teacher and student models, and the enriched input format.