A APPENDIX

A.1 APPARATUS

The stimuli are presented on a desktop computer that has a 27-inch monitor with a resolution of 2560×1440 pixels and a refresh rate of 60 Hz. Participants are required to use the keyboard to interact with the platform. EEG signals are captured and amplified using a Scan NuAmps Express system (Compumedics Ltd., VIC, Australia) and a 64-channel Quik-Cap (Compumedical NeuroScan). A laptop computer functions as a server to record EEG signals and triggers using Curry8 software. Throughout the experiment, electrode-scalp impedance is maintained under $50k\Omega$, and the sampling rate is set at 1000 Hz.

A.2 PREPROCESS

We preprocess the EEG data using several steps: first, we re-reference all recorded signals offline using the linked-mastoids method to reduce reference bias (Yao et al. (2019)); second, we apply notch, high-pass, and low-pass filters to eliminate environmental interference, slow voltage drift, and high-frequency noise respectively; third, we extract epochs of interest and compute their averages to obtain ERP waveforms. The epochs are defined from 200 ms before the presentation of each stimulus word to 800 ms after, covering the expected time window for relevant neural responses.

A.3 ERP ANALYSIS

Table 3: The statistical significance test results (F score) for different ERP components across brain regions for HalluWrong vs. NoHallu words.

F[1,26]	pre-frontal	frontal	central	l-temporal	r-temporal	parietal	occipital
50-120 ms	1.0762	2.2069	0.1047	0.0786	0.5500	1.5207	3.3269
120-280 ms	1.6805	3.3903	0.5743	1.1759	0.0202	3.3387	3.6863
280-550 ms	0.0173	0.0340	0.3558	0.2935	0.1930	0.5038	2.9955
550-750 ms	0.0556	0.0056	1.4622	0.1463	0.1536	3.0617	1.0923

Table 4: The statistical significance test results (p value) for different ERP components across brain regions for HalluWrong vs. NoHallu words.

p value	pre-frontal	frontal	central	l-temporal	r-temporal	parietal	occipital
50-120 ms	0.3095	0.1499	0.7489	0.7816	0.4652	0.2290	0.0801
120-280 ms	0.2067	0.0775	0.4556	0.2885	0.8881	0.0796	0.0750
280-550 ms	0.8963	0.8553	0.5562	0.5928	0.6642	0.4844	0.0958
550-750 ms	0.8155	0.9407	0.2379	0.7053	0.6984	0.0924	0.3060

Tables 3 and Table 4 present the statistical significance test results (F scores and p values, respectively) for different ERP components across brain regions, comparing HalluWrong vs. NoHallu words. The results indicate that for all ERP components and in all examined regions of interest, the differences between HalluWrong and NoHallu words are not statistically significant.

A.4 MODELS

The model structures and hyperparameters are as follows. For all models, the input features first undergo a preprocessing pipeline, which includes mean imputation for any missing values, followed by standard scaling to normalize the data.

SVM (Support Vector Machine) We use a Radial Basis Function (RBF) kernel. The regularization parameter C is set to 1. The model is configured to output probability estimates for classification.

Table 5: The recall results of word-level and sentence-level prediction. † indicates the result is significantly different with p-value$<$0.05 compared to the best model. * indicates including HalluWrong words in the training.

Models	word-level		sentence-level	
	$Recall_{within}$	$Recall_{cross}$	$Recall_{within}$	$Recall_{cross}$
SVM	0.5740†	0.3669†	0.6610†	0.4655†
RF	0.2629†	0.1169†	0.0404†	0.0162†
GBDT	0.4104†	0.2132†	0.4602†	0.0346†
MLP	0.6141†	0.3487†	0.8952†	**0.8244**
attention	**0.6617**	**0.4421**	**0.9310**	0.7407†
SVM*	0.1370†	0.2340†	0.2771†	0.2349†

RF (Random Forest) We set the number of trees in the forest to 100. All other parameters are kept at their default values as specified in the scikit-learn library.

GBDT (Gradient Boosting Decision Trees) We set the number of boosting stages to 100 and the learning rate to 0.1. All other parameters are set to their default values.

MLP (Multi-Layer Perceptron) We implement a network using PyTorch. The architecture consists of a single hidden layer with 100 units, which uses a ReLU activation function. A dropout layer with a probability of 0.5 is applied after the activation function for regularization. The output layer is a linear layer that maps to the two output classes.

Attention-based model We use a Transformer Encoder architecture implemented in PyTorch. The input features are first projected into an embedding space with a dimension of 128. This is followed by a 2-layer Transformer Encoder. Each encoder layer utilizes a multi-head attention mechanism with 8 attention heads and a dropout rate of 0.5. A final linear layer maps the encoder's output to the class scores.

Training Configuration for Deep Models For both deep learning models (MLP and Attention-based), we use the cross-entropy loss function and the Adam optimizer with a learning rate of 10^{-3}. The models are trained for 300 epochs with a batch size of 32.

A.5 More Results

Table 5 shows the recall results of word-level and sentence-level prediction.

A.6 The Use of Large Language Models

In this work, we leveraged large language models (LLMs) to assist in manuscript preparation, including refining the text for clarity and style, as well as facilitating literature retrieval. All LLM-generated suggestions were carefully reviewed, edited, and integrated by the authors to ensure scientific accuracy and consistency with our own writing voice. Meanwhile, the dataset on which our experiments depend was generated using an open-source multimodal large language model (MLLM), i.e., Qwen2.5-VL-3B. We adopted Qwen2.5-VL-3B to produce image-based descriptions from which we selected hallucinated and non-hallucinated content stimuli for our EEG experiment. We acknowledge the ongoing discourse around the ethical use of LLMs in scholarly writing—particularly regarding transparency, originality, and accountability. We transparently report the use of LLM assistance and reaffirm that all substantive intellectual contributions (e.g. experimental design, data analysis, interpretation) originated from the authors.