# Appendix

## A. Deep Neural Networks' Activations

**Pre-trained tansformers** In Section 4, we extract the activations of GPT-2 (Radford et al., 2019) and five transformer architectures: BERT (Devlin et al., 2019), XLnet (Yang et al., 2020), Roberta (Liu et al., 2019), AlBert (Lan et al., 2020) and DistilGPT-2. We use the pre-trained models from Huggingface (Wolf et al., 2020): 'bert-base-cased', 'xlnet-base-cased', 'roberta-base', 'albert-base-v1', and 'distilGPT-2' respectively. In Figure 7, we focus on one middle layer of these transformers ($l = n_{\text{layers}} \times 2/3$), because it has shown to best encode brain activity (Caucheteux & King, 2020) and to encode relevant linguistic properties (Manning et al., 2020; Jawahar et al., 2019).

**Text formatting and tokenization** To extract the activations elicited by one story, we proceed as follows: we first format and lower case the text (replacing special punctuation marks such as "–" and duplicated marks "?." by dots), then apply the tokenizer provided by Huggingface (Wolf et al., 2020) to convert the transcript into either word-level or sub-word-level tokens called "Byte Pair Encoding" (BPE) (Sennrich et al., 2016). Here, more than 99.5% of BPE-level tokens were complete words. The tokens are then split into sections of 256 tokens (this length is constrained by GPT-2's architecture) and input to the deep network one story at a time. The activations of each layer are finally extracted, resulting in $n_{\text{layers}}$ vectors of 768 activations for each token of each story transcript. In the 0.5% case where BPE are not complete words, BPE-features are summed between successive words, to obtain $n_{\text{layers}}$ vectors per word per story.

## B. Convergence of the Method to Build $\overline{X}$

In Section 3.1 and 3.3, we compute the syntactic component $\overline{X}$ of GPT-2 activations $X$ elicited by a sentence $w$. $\overline{X}$ is approximated by $\overline{X_k}$, the average activations across $k$ sentences with the same syntax as $w$. Here, we sample $k = 10$ sentences. We check in Figure 8 that the method has converged before $k = 10$. We compute the cosine similarity between $\overline{X}_k$ and $\overline{X}_{k-1}$ for $k$ between 1 and 15. The syntactic embeddings stabilize with at least eight sampled sentences.

## C. Evaluating the Level of Semantic and Syntactic Information in $\overline{X}$

In Section 3.3 and Figure 3, we check that the syntactic embedding $\overline{X}$ extracted from GPT-2 only contains syntax. To this aim, we evaluate the ability of a linear decoder to predict two syntactic features and three semantic features from $\overline{X}$.

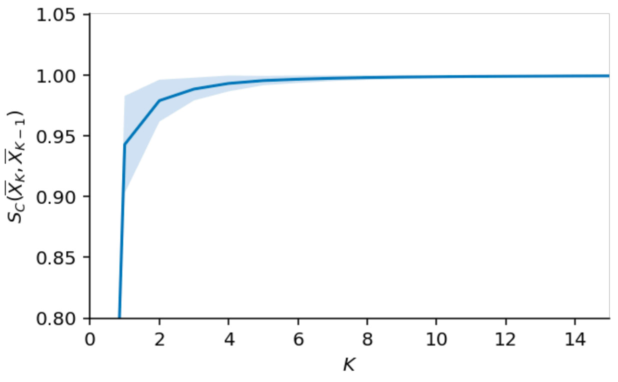

*Figure 8.* **Convergence of the method to build syntactic embeddings.** Cosine similarity $S_C$ between the syntactic component $\overline{X}$ of GPT-2 activations induced by a sequence $w$, when computed with $K$ and $K-1$ syntactically equivalent sequences. The syntactic embeddings $\overline{X}_K$ and $\overline{X}_{K-1}$ are computed for 100 Wikipedia sentences ($\approx 2,800$ words), and the similarity scores are averaged across embeddings. In shaded, the 95% confidence interval across embeddings.

**Semantic and syntactic features** The two syntactic features derived from the stimulus are:

- The part-of-speech of the words (categorical feature), as defined by Spacy tags (Honnibal et al., 2020).
- The depth of the syntactic tree (continuous feature). The syntactic tree is extracted with the state-of-the-art Supar dependency parser (Zhang et al., 2020).

The three semantic features are only computed for verbs, nouns and adjectives (as defined by Spacy part-of-speech tags) and are the followings:

- Word frequency (labeled as 'Word freq' in Figure 3, continuous feature). We use the 'zipf_frequency' from the wordfreq[4] python library.
- Word embedding (continuous feature), computed using the pre-trained model from Spacy (Honnibal et al., 2020) ('en_core_web_lg', 300 dimensions).
- Semantic category (categorical feature). We used the 47 semantic categories[5]. Categories are not available

[4] https://pypi.org/project/wordfreq/

[5] Categories are: abstract, action, animal, auditory, body, building, cognitive, construct, creative, device, distant, document, electronic, emotion, emotional, entity, event, food, furniture, general, geological, group, human, instrument, locative, mental, miscel-

for all the 2,800 Wikipedia words studied here. Thus, we first train a linear model (scikit-learn 'RidgeCV-Classifier') to predict the semantic category of the 535 labeled words used in (Binder et al., 2016), given their Spacy word embedding (300 dimensions). We then label the 2,800 Wikipedia words using the semantic category predicted by the classifier.

**Linear decoder** To evaluate the ability of a linear decoder to predict the five linguistic features from $\overline{X}$, we:

- Build syntactic embeddings $\overline{X}$ for 100 Wikipedia sentences ($\approx 2,800$ words), following Section 3.1, using the ninth layer of GPT-2.
- Build the three semantic and two syntactic features described above from the 2,800 Wikipedia words Wikipedia words.
- Fit a $\ell_2-$regularized linear model to predict the five features given the syntactic embeddings. We use the 'RidgeCV' regressor (resp. 'RidgeClassifierCV' classifier) from scikit-learn (Pedregosa et al., 2011) to predict the continuous (resp. categorical) features, with ten possible penalization values log-spaced between $10^{-3}$ and $10^6$.
- Evaluate the linear model on held out data, using a 10 cross-validation setting ('KFold' cross-validation from scikit-learn). Performance is assessed using *adjusted* accuracy ('balanced_accuracy_score' from scikit-learn) for the categorical features, and $R^2$ for the continuous features. Thus, the chance level is zero for both types of features, and the best score is one.
- Report the average decoding performance in Figure 3 (red bars), and the standard-error of the means across the ten test folds.

For comparison, we repeat the exact same procedure with the full GPT-2 activations $X$ (instead of their syntactic component $\overline{X}$), and report the results in Figure 3 (grey bars).

## D. Temporal Alignment $g$ between $X$ and $Y$

In Section 3.2, we map the network's activations $X$ (of length $M$, the number of words) and the brain response $Y$ (of length $N$, the number of fMRI measurements) induced by the same story $w$ (of $M$ words). $M$ is usually greater than $N$. To align the two spaces, we first sum the features between successive fMRI measurements, and then apply a finite impulse response (FIR) model. We denote $g$ this transformation. Specifically, for each fMRI time sample $i \in [1 \dots N]$, $g_i$ combines word features within each acquisition interval as follows:

$$g_i : \mathbb{R}^{M \times d} \to \mathbb{R}^{5d}$$
$$u \mapsto \left[\widetilde{u_i}, \widetilde{u_{i-1}}, \dots, \widetilde{u_{i-4}}\right]$$
$$\widetilde{u_i} = \sum_{\substack{m \in [\![1 \dots M]\!] \\ \mathcal{T}(m)=i}} u_m$$

with

$$\mathcal{T} : [\![1 \dots M]\!] \to [\![1 \dots N]\!]$$
$$m \mapsto i \quad / \quad |t_{y_i} - t_{x_j}| = \min_{k \in [\![1 \dots N]\!]} |t_{y_k} - t_{x_m}|$$

with $\tilde{u}$ the summed activations of words between successive fMRI time samples, $u$ the five lags of FIR features, $(t_{x_1}, \dots, t_{x_M})$ the timings of the $M$ words onsets, and $(t_{y_1}, \dots, t_{y_N})$ the timings of the $N$ fMRI measurements.

## E. Brain Parcellation

In Figure 6, brain scores are averaged across voxels within regions of interest using the Brodmann's areas from the PALS parcellation of freesurfer[6]. To gain in precision, we split the superior temporal gyrus (BA22) into its anterior, middle and posterior parts. In Figure 6, we report the top ten areas of the left hemisphere in term of average brain score. Certain areas are renamed for clarity, as specified in the table below:

| Label | Corresponding Brodmann's areas |
|---|---|
| A1 | BA41 / BA42 |
| Fusiform | BA37 |
| Angular | BA39 |
| aSTG | BA22-anterior |
| mSTG | BA22-middle |
| pSTG | BA22-posterior |
| M1 | BA4 |
| Supramarginal | BA40 |
| IFG (Op) | BA44 |
| IFG (Tri) | BA45 |
| IFG (Orb) | BA47 |
| Middle-frontal | BA46 |
| V1 | BA17 |
| Fronto-polar | BA10 |
| Temporo-polar | BA38 |
| Precuneus | BA7 |
| Cingulate | BA23 / BA26 / BA29 / BA30 / BA31 |

laneous, multimodal, object, part, perceptual, period, physical, place, plant, property, social, somatosensory, sound, spatial, state, temporal, time, tool, vehicle, visual, weather

[6]https://surfer.nmr.mgh.harvard.edu/fswiki/PALS_B12

## F. Control for Low-level Linguistic Features

In Section 5 and Figure 7, we check that the brain scores are not driven by low-level linguistic features. Thus, we compute the $R$ scores of GPT-2 activations (ninth layer) induced by modified versions of the stimulus:

- Random words sampled from the same story. Words are uniformly sampled from the words of the story, tokenized using Spacy (Honnibal et al., 2020). Punctuation marks are considered as words. Upper-cases are kept.
- Random sentences from Wikipedia, of the same length as the sentences of the stimulus. We first build a dictionary of (length, list of match-length sentences) pairs out of 10K sentences from Wikipedia ($\approx$ 577K words). Then, for each sentence of the stimulus, a sentence is uniformly sampled from the set of Wikipedia match-length sentences.
- The sentences of the stimulus, but with random word order. Words are shuffled *within* each sentence.

Then, we extract the corresponding GPT-2 activations and compute the $R$ scores following Section 4. $R$ scores are evaluated for each subject and reported in Figure 7.