A Appendix / supplemental material

A.1 Supplemental Methods

A.1.1 fMRI preprocessing

The text of the following sections (Preprocessing of B0 inhomogeneity mappings, Anatomical data preprocessing, Functional data preprocessing) was automatically generated by fMRIPrep with the express intention that users should copy and paste this text into their manuscripts unchanged. It is released under the CC0 license.

Preprocessing of B0 inhomogeneity mappings A total of 1 fieldmaps were found available within the input BIDS structure for this particular subject. A B0-nonuniformity map (or fieldmap) was estimated based on two (or more) echo-planar imaging (EPI) references with topup (Andersson, Skare, and Ashburner (2003); FSL 6.0.5.1:57b01774).

Anatomical data preprocessing A total of 1 T1-weighted (T1w) images were found within the input BIDS dataset.The T1-weighted (T1w) image was corrected for intensity non-uniformity (INU) with N4BiasFieldCorrection [50], distributed with ANTs 2.3.3 [51], and used as T1w-reference throughout the workflow. The T1w-reference was then skull-stripped with a Nipype implementation of the antsBrainExtraction.sh workflow (from ANTs), using OASIS30ANTs as target template. Brain tissue segmentation of cerebrospinal fluid (CSF), white-matter (WM) and gray-matter (GM) was performed on the brain-extracted T1w using fast [52]. Brain surfaces were reconstructed using recon-all (FreeSurfer 6.0.1 [53]), and the brain mask estimated previously was refined with a custom variation of the method to reconcile ANTs-derived and FreeSurfer-derived segmentations of the cortical gray-matter of Mindboggle [54]. Volume-based spatial normalization to two standard spaces (MNI152NLin2009cAsym, MNI152NLin6Asym) was performed through nonlinear registration with antsRegistration (ANTs 2.3.3), using brain-extracted versions of both T1w reference and the T1w template. The following templates were selected for spatial normalization: ICBM 152 Nonlinear Asymmetrical template version 2009c [55] [TemplateFlow ID: MNI152NLin2009cAsym], FSL's MNI ICBM 152 non-linear 6th Generation Asymmetric Average Brain Stereotaxic Registration Model [56][TemplateFlow ID: MNI152NLin6Asym].

Functional data preprocessing For each of the 7 BOLD runs found per subject (across all tasks and sessions), the following preprocessing was performed. First, a reference volume and its skull-stripped version were generated by aligning and averaging 1 single-band references (SBRefs). Head-motion parameters with respect to the BOLD reference (transformation matrices, and six corresponding rotation and translation parameters) are estimated before any spatiotemporal filtering using mcflirt (FSL 6.0.5.1:57b01774 [57]). The estimated fieldmap was then aligned with rigid-registration to the target EPI (echo-planar imaging) reference run. The field coefficients were mapped on to the reference EPI using the transform. BOLD runs were slice-time corrected to 0.7s (0.5 of slice acquisition range 0s-1.4s) using 3dTshift from AFNI [58]. The BOLD reference was then co-registered to the T1w reference using bbregister (FreeSurfer) which implements boundary-based registration [59]. Co-registration was configured with six degrees of freedom. First, a reference volume and its skull-stripped version were generated using a custom methodology of fMRIPrep. Several confounding time-series were calculated based on the preprocessed BOLD: framewise displacement (FD), DVARS and three region-wise global signals. FD was computed using two formulations following Power (absolute sum of relative motions [60]) and Jenkinson (relative root mean square displacement between affines [57]). FD and DVARS are calculated for each functional run, both using their implementations in Nipype (following the definitions by [60]). The three global signals are extracted within the CSF, the WM, and the whole-brain masks. Additionally, a set of physiological regressors were extracted to allow for component-based noise correction (CompCor [61]). Principal components are estimated after high-pass filtering the preprocessed BOLD time-series (using a discrete cosine filter with 128s cut-off) for the two CompCor variants: temporal (tCompCor) and anatomical (aCompCor). tCompCor components are then calculated from the top 2% variable voxels within the brain mask. For aCompCor, three probabilistic masks (CSF, WM and combined CSF+WM) are generated in anatomical space. The implementation differs from that of Behzadi et al. [61] in that instead of eroding the masks by 2 pixels on BOLD space, the aCompCor masks are subtracted a mask of pixels that likely contain a volume fraction of GM. This

mask is obtained by dilating a GM mask extracted from the FreeSurfer's aseg segmentation, and it ensures components are not extracted from voxels containing a minimal fraction of GM. Finally, these masks are resampled into BOLD space and binarized by thresholding at 0.99 (as in the original implementation). Components are also calculated separately within the WM and CSF masks. For each CompCor decomposition, the k components with the largest singular values are retained, such that the retained components' time series are sufficient to explain 50 percent of variance across the nuisance mask (CSF, WM, combined, or temporal). The remaining components are dropped from consideration. The head-motion estimates calculated in the correction step were also placed within the corresponding confounds file. The confound time series derived from head motion estimates and global signals were expanded with the inclusion of temporal derivatives and quadratic terms for each [62]. Frames that exceeded a threshold of 0.5 mm FD or 1.5 standardised DVARS were annotated as motion outliers. The BOLD time-series were resampled into standard space, generating a preprocessed BOLD run in MNI152NLin2009cAsym space. First, a reference volume and its skull-stripped version were generated using a custom methodology of fMRIPrep. The BOLD time-series were resampled onto the following surfaces (FreeSurfer reconstruction nomenclature): fsaverage. Grayordinates files [63] containing 170k samples were also generated using the highest-resolution fsaverage as intermediate standardized surface space. All resamplings can be performed with a single interpolation step by composing all the pertinent transformations (i.e. head-motion transform matrices, susceptibility distortion correction when available, and co-registrations to anatomical and output spaces). Gridded (volumetric) resamplings were performed using antsApplyTransforms (ANTs), configured with Lanczos interpolation to minimize the smoothing effects of other kernels [64]. Non-gridded (surface) resamplings were performed using mri_vol2surf (FreeSurfer).

Many internal operations of fMRIPrep use Nilearn 0.8.1[65], mostly within the functional processing workflow. For more details of the pipeline, see the section corresponding to workflows in fMRIPrep's documentation.

Data was smoothed with a 3mm FWHM kernel for subsequent localizer and encoding model analyses. Data was smoothed with a 6mm FWHM kernel for computing the intersubject correlation mask, which is in the recommended smoothing range [41].

A.1.2 Sparse random projection

Sparse random projection projects a high dimensional feature space into a lower dimensionality feature space while preserving the pairwise Euclidean distance between points. The dimensionality of the lower dimensional space is determined using the Johnson-Lindenstrauss lemma and an epsilon specifying the amount of tolerated distortion [36]. Using the standard epsilon value of 0.1 and our sample size of 1921 time points, the Johnson-Lindenstrauss lemma outputs a target dimensionality of 6480 projections. These projections are randomly generated as a sparse matrix of nearly orthogonal dimensions. The feature spaces are projected onto this matrix using the dot product. The result is a 1921 x 6480 dimensional feature space, which is then used to predict neural activity in the encoding model. This pipeline has been used in several recent papers to speed up model fitting and to avoid overfitting [29, 47].

A.2 Supplemental figures

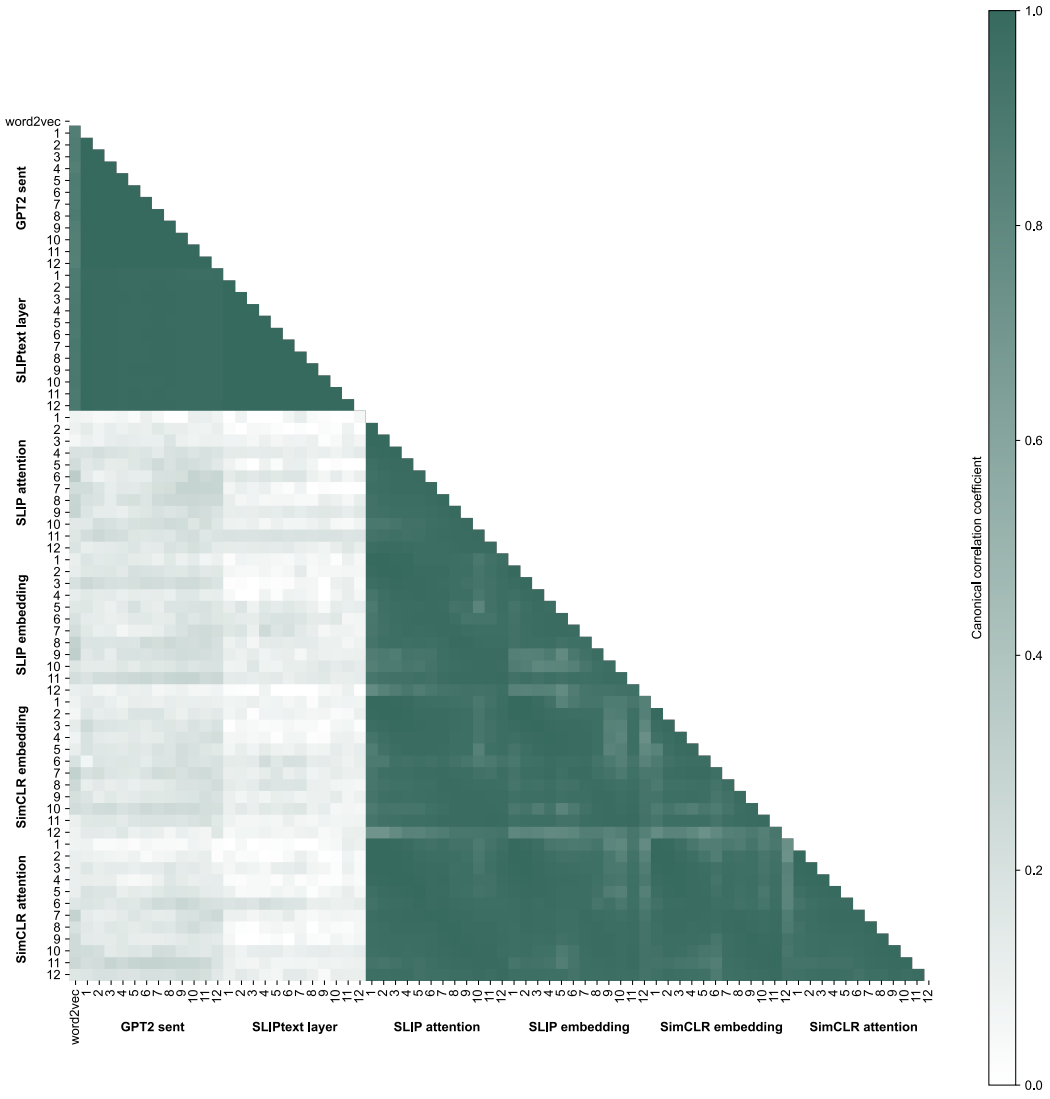

Figure 8: Similarity between vision and language model representations during a naturalistic movie.

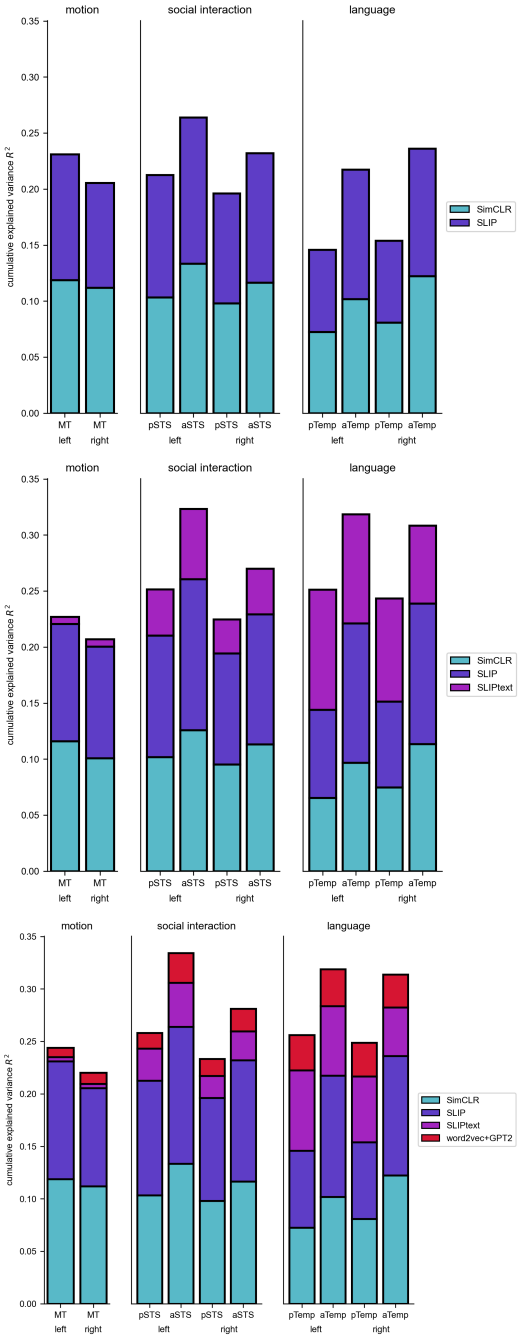

Figure 9: **Top**: Predictive contribution of features in an encoding model with vision embeddings from SimCLR and SLIP. **Middle**: Predictive contribution of features in an encoding model with vision embeddings from SimCLR and SLIP and language embeddings from SLIP's language encoder. **Bottom**: Predictive contribution of features in an encoding model with vision embeddings from SimCLR and SLIP and language embeddings from SLIP's language encoder, word2vec, and GPT-2 (reproduction of figure 4 from main text for visualization here). Each rectangle represents the variance explained by that feature space (all layers are added together when relevant), and averaged across participant-defined regions of interest.

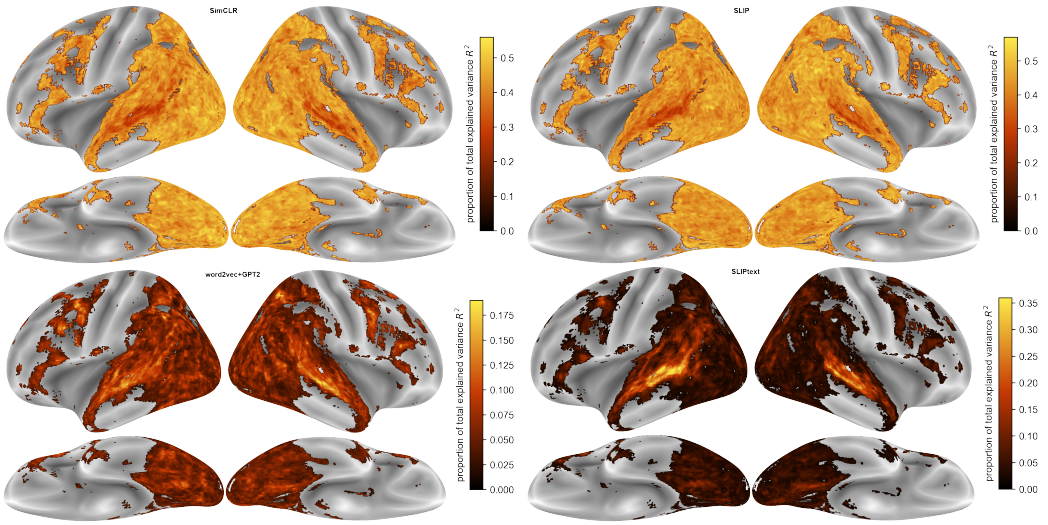

Figure 10: Group maps of the proportion of total variance explained by all layers of SimCLR, all layers of SLIP, all layers of SLIPtext, and all layers of GPT-2 + word2vec. All maps thresholded at 0.01.

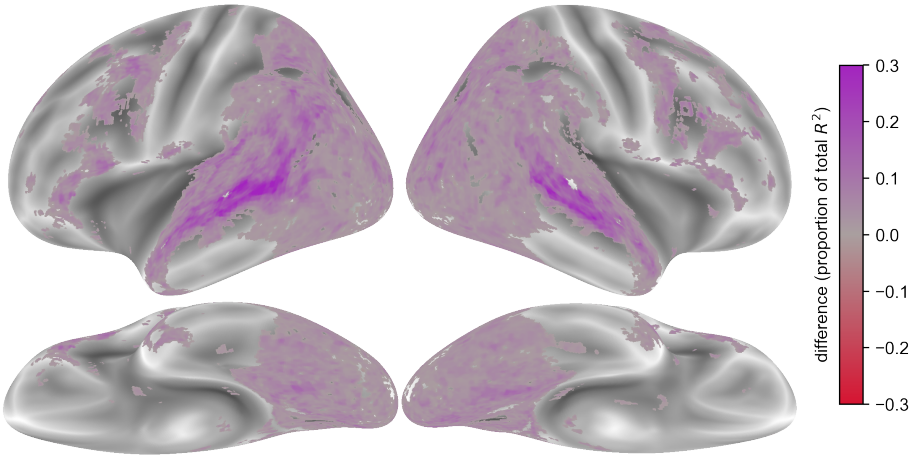

Figure 11: Group map of the difference in proportion of variance explained between SLIP's language encoder and word2vec in the full model, thresholded at difference of 0.01. Red indicates where word2vec explains more variance than SLIPtext and purple indicates where SLIPtext explains more variance than word2vec. Thresholded at 0.01.

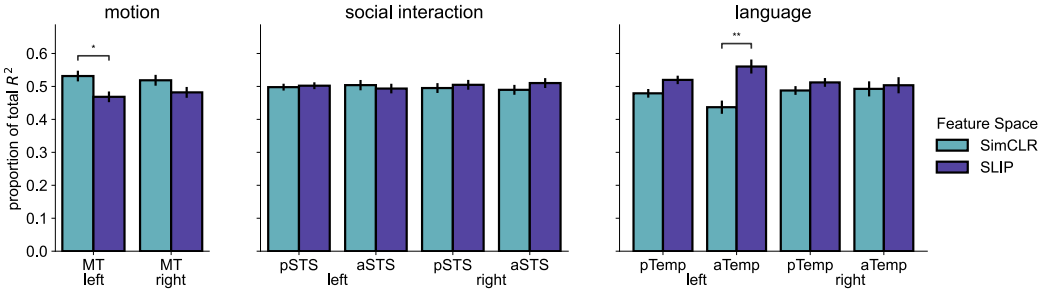

Figure 12: Proportion of variance explained by SimCLR and SLIP's vision embeddings when fit in one encoding model.

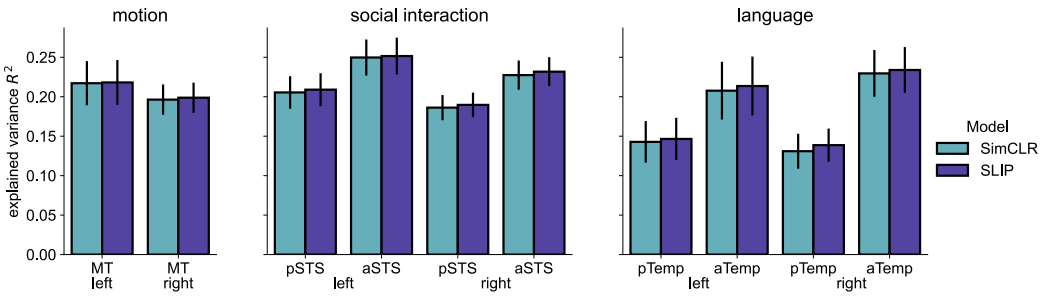

Figure 13: Encoding model performances of vision embeddings of just SimCLR and vision embeddings of just SLIP.

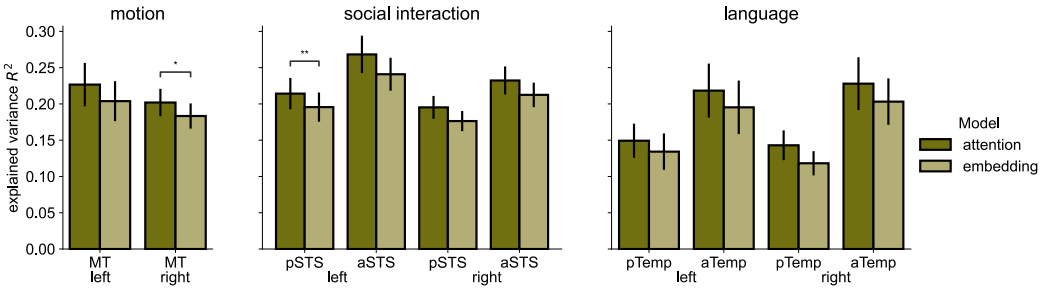

Figure 14: Performance of all attention head output and embeddings from SimCLR and SLIP.

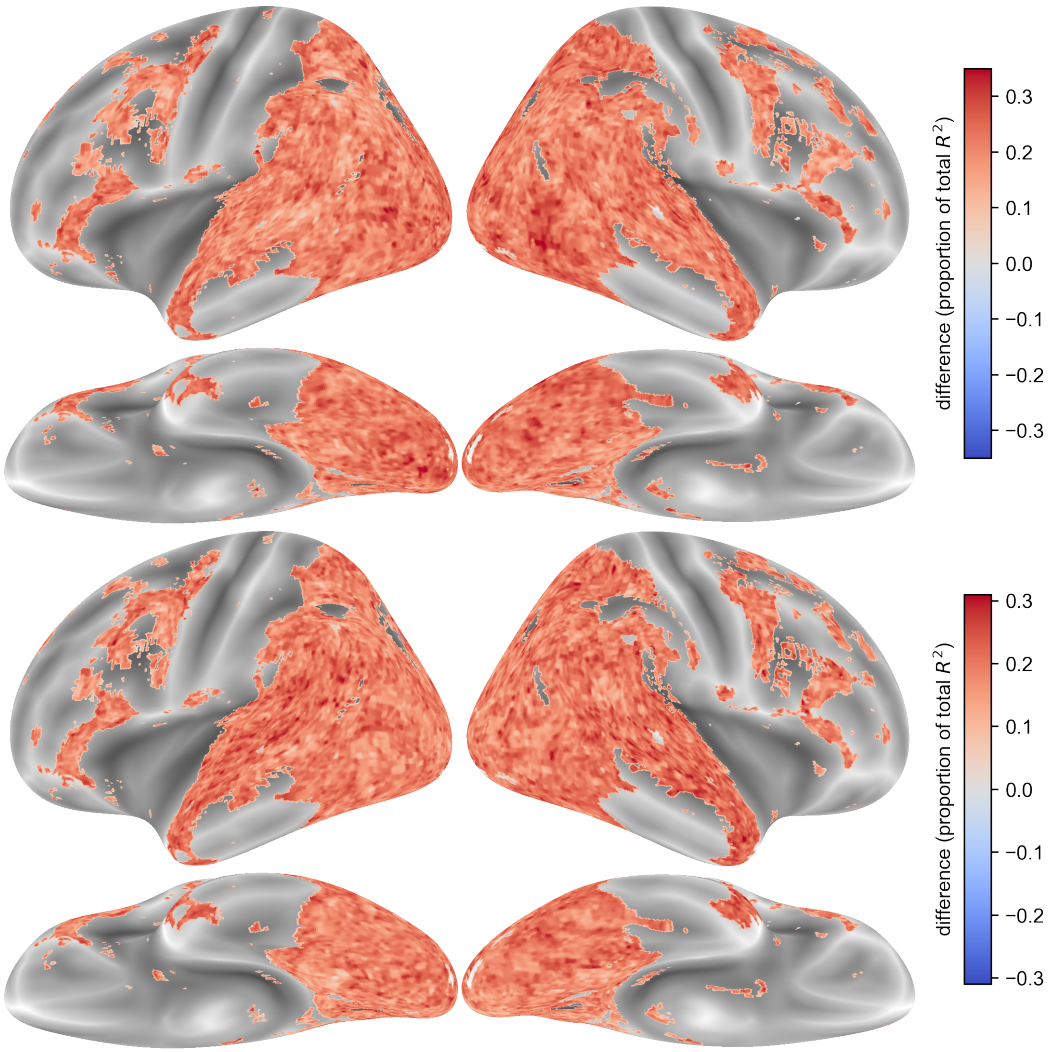

Figure 15: Group maps of the difference between all attention and all embedding layers in SimCLR (top) and SLIP (bottom). Pink indicates where attention predicts better than embeddings and green indicates where embedding predicts better than embeddings.