Similarity Analysis of Contextual Word Representation Models

Wu+ ACL 2020

Motivation

- There are many contextual word representations proposed based on different architectures.
    - ElMo, BERT, XLNet, GPT, Albert, RoBERTa, ...
- How are these different empirically?
- This paper proposes various methods to measure the similarity between the models trained using different architectures.
- This is different from probing where models are compared based on their performance on different downstream (or probing) tasks
    - We need linguistic annotations for probing
    - Comparisons between models are indirect (via probing accuracy)
- Research questions addressed in this paper
    - Do different models behave similarly on the same inputs?
    - Which design choices determine whether models behave similarly or differently?
    - Are certain model components more similar than others across architectures?
    - Is information in a model is localised (i.e. one computing element for each represented entity)

Findings

- Different architectures have rather similar representations, but different individual neurones.

- Lower layers are more similar than higher layers across architectures.

- Higher layers have more localised representations than lower layers.

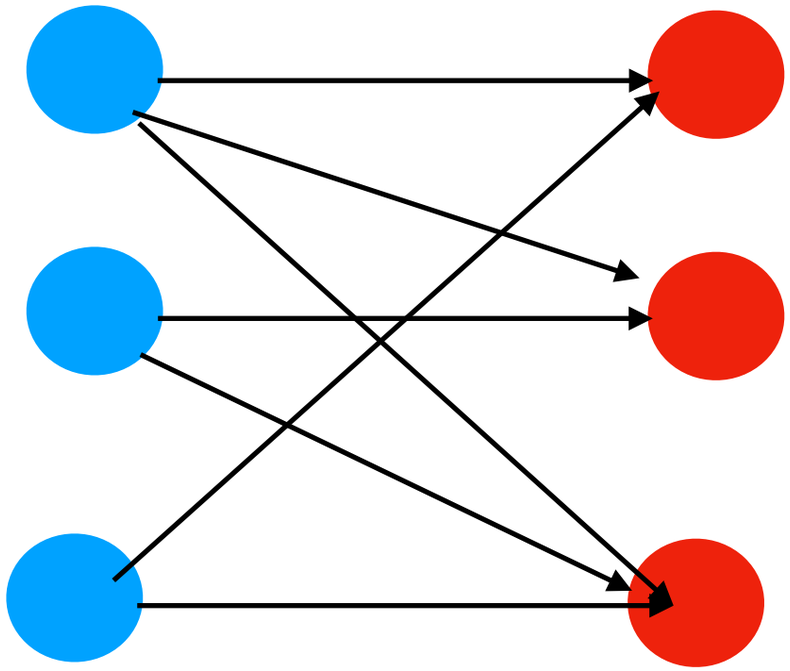
- Higher layers are more affected by fine-tuning on downstream tasks than lower layers.

- Fine-tuning affects the localisation of information, causing high layers to be less localised.

Similarity Measures

- Collection of M models $\{f^{(m)}\}_{m=1}^{M}$
- word representations $\mathbf{h}_l^{(m)}$ by the m-th model in its l-th layer
- attention weights $\alpha_l^{(m)}$
- Neurones are indexed by k, $\mathbf{h}_l^{(m)}[k]$ and $\alpha_l^{(m)}[k]$
- In the case of word embeddings [k] denotes the k-th dimension and for attention weights [k] will be the value of the k-th attention head (possibly a matrix)
- Various similarity measures are proposed between neurones, $\texttt{sim}(\mathbf{h}_l^{(m)}, \mathbf{h}_{l'}^{(m')})$ and between attentions $\texttt{sim}(\alpha_l^{(m)}, \alpha_{l'}^{(m')})$

Neurones vs. Dimensions



hidden layer
dimensions correspond to neurones

We are not interested in the edges (weights) in this analysis

Neurone-level Similarity

Compare one neurone from a layer of one model against another neurone from a layer in a different model.

For a given neuron $\mathbf{h}_l^{(m)}[k]$, we define

$$\texttt{neuronsim}(\mathbf{h}_l^{(m)}[k], \mathbf{h}_{l'}^{(m')}) = \max_{k'} |\rho(\mathbf{h}_{l'}^{(m')}[k'], \mathbf{h}_l^{(m)}[k])|$$

as the maximum correlation between it and another neuron in some layer (Bau et al., 2019). Here ρ is the Pearson correlation. This naturally gives rise to an aggregate measure at the layer level:

$$\texttt{neuronsim}(\mathbf{h}_l^{(m)}, \mathbf{h}_{l'}^{(m')}) = \frac{1}{N_m} \sum_k \texttt{neuronsim}(\mathbf{h}_l^{(m)}[k], \mathbf{h}_{l'}^{(m')})$$

Mixed neurone-representation similarity

Information of one neurone can be distributed across neurones in another layer. Do least square regression and find the minimum regression score.

We define

$$\texttt{mixedsim}(\mathbf{h}_l^{(m)}[k], \mathbf{h}_{l'}^{(m')}) :=$$

$$\text{lstsq}(\mathbf{h}_{l'}^{(m')}, \mathbf{h}_l^{(m)}[k])\,\texttt{.r}$$

where `.r` is the r-value associated with the regression, the norm of the prediction divided by the norm of the regressand. As before, this is extended to the layer level:

$$\texttt{mixedsim}(\mathbf{h}_l^{(m)}, \mathbf{h}_{l'}^{(m')}) =$$

$$\frac{1}{N_m} \sum_k \texttt{mixedsim}(\mathbf{h}_l^{(m)}[k], \mathbf{h}_{l'}^{(m')})$$

Representation-level Similarity

- A representation-level measure find correlations between a full model (or layer) simultaneously.
  - Canonical Correlation Analysis (CCA) [Raghu+17]
  - Projection weighted CCA [Kornblith+19]

SVCCA Given two layers

$$\mathbf{X}, \mathbf{Y} = \mathbf{Zh}_{l_x}^{(m_x)}, \mathbf{Zh}_{l_y}^{(m_y)}$$

we compute the truncated principal components

$$\mathbf{X}', \mathbf{Y}' = \mathbf{U_x}[:, : l_x], \mathbf{U_y}[:, : l_y]$$

where $\mathbf{U_x}$ are the left singular vectors of $\mathbf{X}$, and l_x is the index required to account for 99% of the variance. $\mathbf{U_y}$ and l_y are defined analogously. The SVCCA correlations, ρ_{SVCCA}, are defined as:

$$\mathbf{u}, \rho_{SVCCA}, \mathbf{v} = \text{SVD}(\mathbf{X'}^T \mathbf{Y}')$$

The SVCCA similarity, $\texttt{svsim}(\mathbf{h}_{l_x}^{(m_x)}, \mathbf{h}_{l_y}^{(m_y)})$, is the mean of ρ_{SVCCA}.

PWCCA Identical to SVCCA, except the computation of similarity is a weighted mean. Using the same notation as above, we define canonical vectors,

$$\mathbf{H_X} := \mathbf{X}'\mathbf{u}$$
$$\mathbf{H_Y} := \mathbf{Y}'\mathbf{v}$$

We define alignments

$$\mathbf{A_X} := \text{abs}\left(\mathbf{H_X^T X}\right)$$
$$\mathbf{A_Y} := \text{abs}\left(\mathbf{H_Y^T Y}\right)$$

where abs is the element-wise absolute value. The weights are

$$\alpha_x := \text{weights}(\mathbf{A_X 1}), \quad \alpha_y := \text{weights}(\mathbf{A_Y 1})$$

where $\mathbf{1}$ is the column vector of all ones, and weights normalizes a vector to sum to 1. The PWCCA similarity is

$$\texttt{pwsim}(\mathbf{h}_{l_x}^{(m_x)}, \mathbf{h}_{l_y}^{(m_y)}) := \alpha_x^T \rho_{SVCCA}$$
$$\texttt{pwsim}(\mathbf{h}_{l_y}^{(m_y)}, \mathbf{h}_{l_x}^{(m_x)}) := \alpha_y^T \rho_{SVCCA}$$

It is asymmetric.

Attention-level Similarity

- Find the correlation between attention heads
  - Norm between the two attention matrices $\alpha_l^{(m)}[k], \alpha_{l'}^{(m')}[k']$
  - Their Pearson correlation
  - Their Jensen-Shanon divergence

We define

$$\texttt{attnsim}(\boldsymbol{\alpha}_l^{(m)}[k], \boldsymbol{\alpha}_{l'}^{(m')}) = \max_{k'} \left[\mathrm{Sim}(\boldsymbol{\alpha}_{l'}^{(m')}[k'], \boldsymbol{\alpha}_l^{(m)}[k])\right]$$

We consider three such values of Sim.

- Matrix norm: for each sentence s_i, compute the Frobenius norm $\left\|\boldsymbol{\alpha}_{l'}^{(m')}[h'](s_i) - \boldsymbol{\alpha}_l^{(m)}[h](s_i)\right\|$. Then average over sentences in the corpus.
- Pearson correlation: for every word x_i, compare the attention distributions the two heads induce from x_i to all words under Pearson correlation: $\rho\left(\boldsymbol{\alpha}_{l',i}^{(m')}[h'], \boldsymbol{\alpha}_{l,i}^{(m)}[h]\right)$. Then average over words in the corpus.
- Jensen–Shannon divergence: for every word x_i, compare the attention distributions under Jensen–Shannon divergence: $\frac{1}{2}\,\mathrm{KL}(\alpha_{l',i}^{(m')}[h'] \;\|\; \beta) + \frac{1}{2}\,\mathrm{KL}(\alpha_{l,i}^{(m)}[h] \;\|\; \beta)$, where KL is the KL-divergence and β is the average of the two attention distributions. Then average of words in the corpus.

Distributed Attention-level Similarity

- Compare entire attention heads in two layers by concatenating all weights from all heads in one layer to get an attention representation.

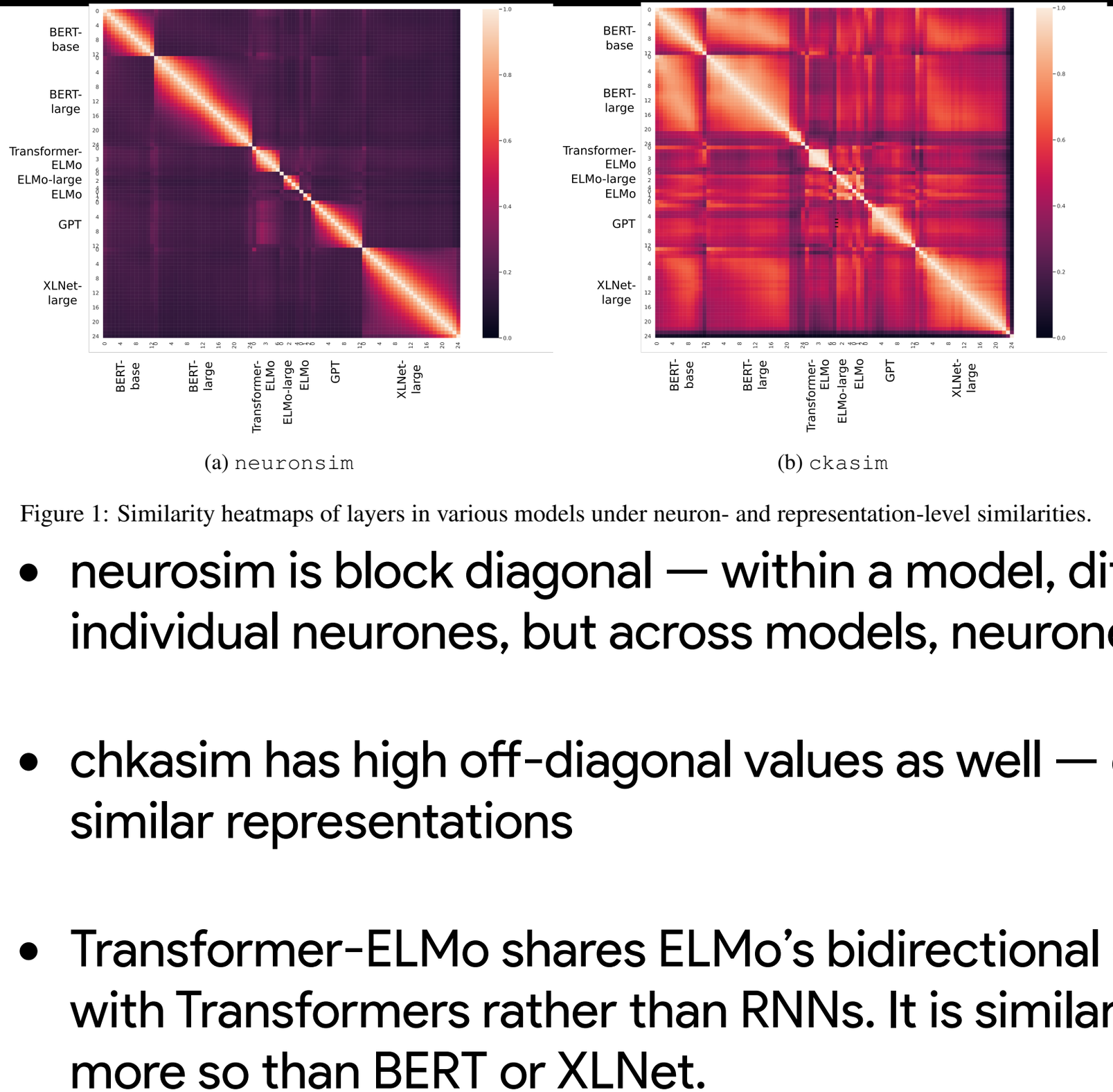
Experimental Setup

- Models
  - ELMo variants: original, bidirectional RNN with 2 hidden layers, 4 layers and Transformer-equivalent variant
  - GPT variants: GPT, GPT-2
  - BERT: base, large
  - XLNet: base, large

Data

- Run on Penn Treebank development and obtain word representations
- Aggregated sub-word representations by taking the representation fr the last sub-word
- Sub-word attentions are computing by summing up attention to sub-words and averaging attention from sub-words (guarantees that attention from each word sums to one)

Similarity of Pre-trained Models



(a) `neuronsim` (b) `ckasim`

Figure 1: Similarity heatmaps of layers in various models under neuron- and representation-level similarities.

ckasim: representation-level similarity based on linear centred kernel alignment

- neurosim is block diagonal — within a model, different layers have similar individual neurones, but across models, neurones are very different.
- chkasim has high off-diagonal values as well — different models generate similar representations
- Transformer-ELMo shares ELMo's bidirectional objective function but with Transformers rather than RNNs. It is similar to both ELMo and GPT, more so than BERT or XLNet.

mixedsim

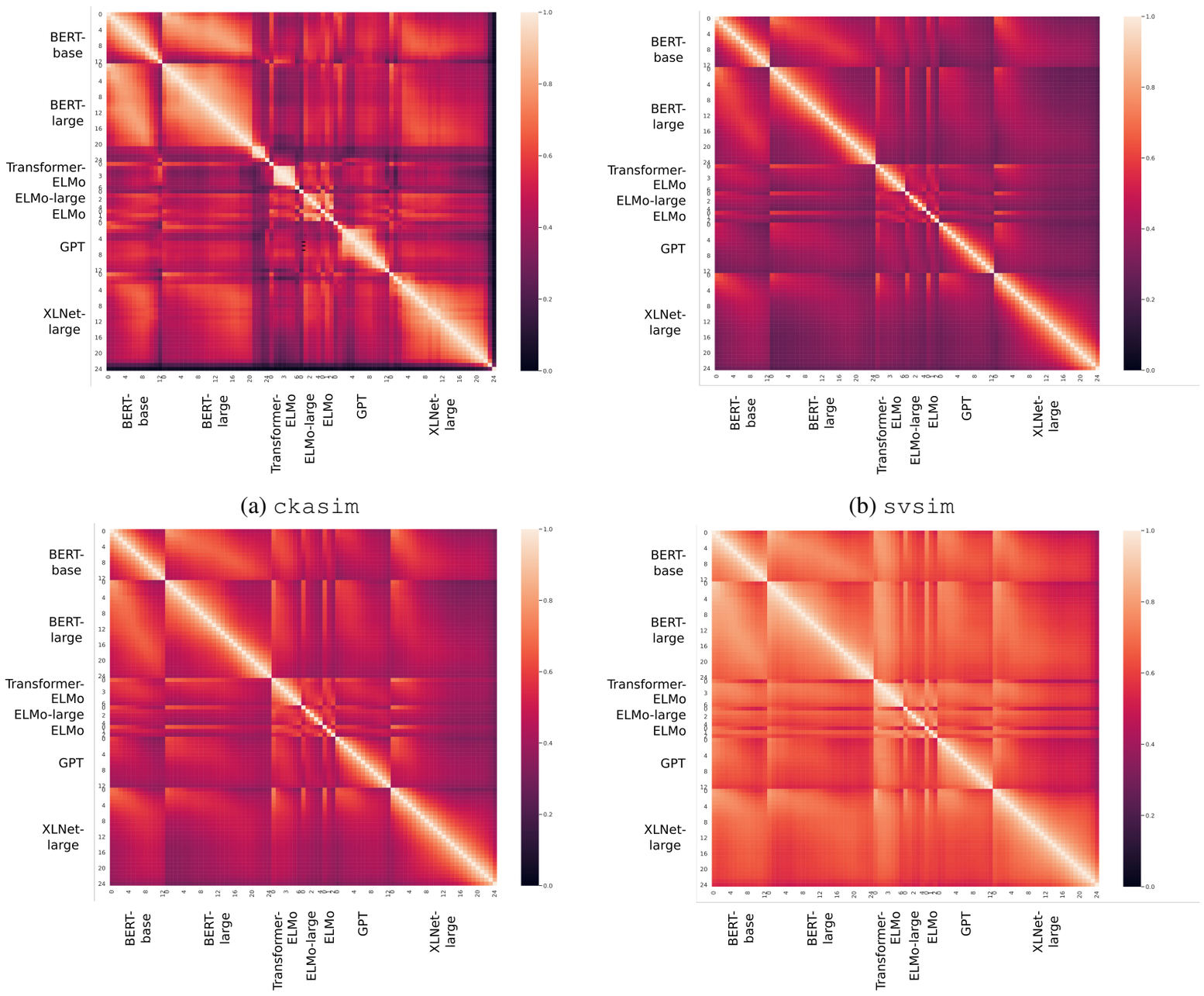


Figure 8: Similarity heatmaps of layers in various models under different representation-level similarity measures.

- Individual neurones in one model may be well represented by a linear combination of neurones in another layer
- Models within the same family are more similar

Why?

- Why are models within the same family are more similar to each other?
  - Could be because models within the sam family are often trained on the same data, but cross-family models are trained on different data.

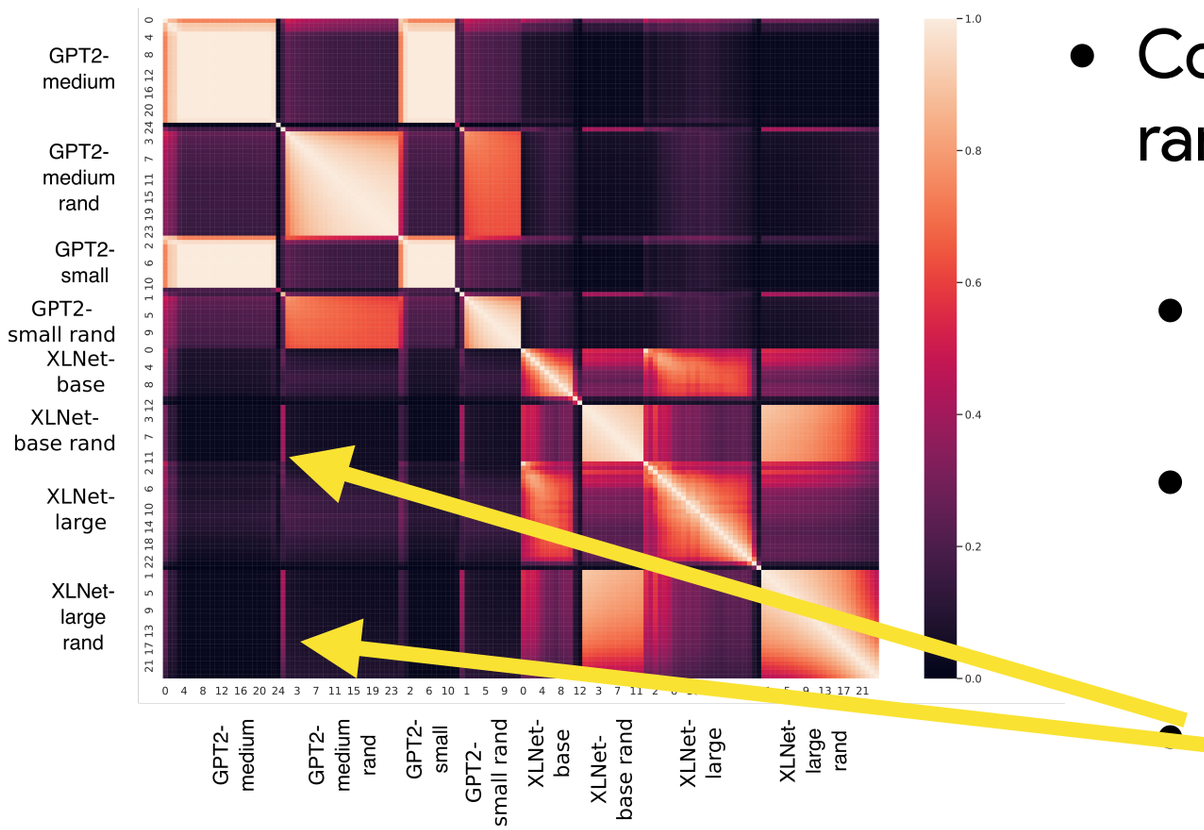
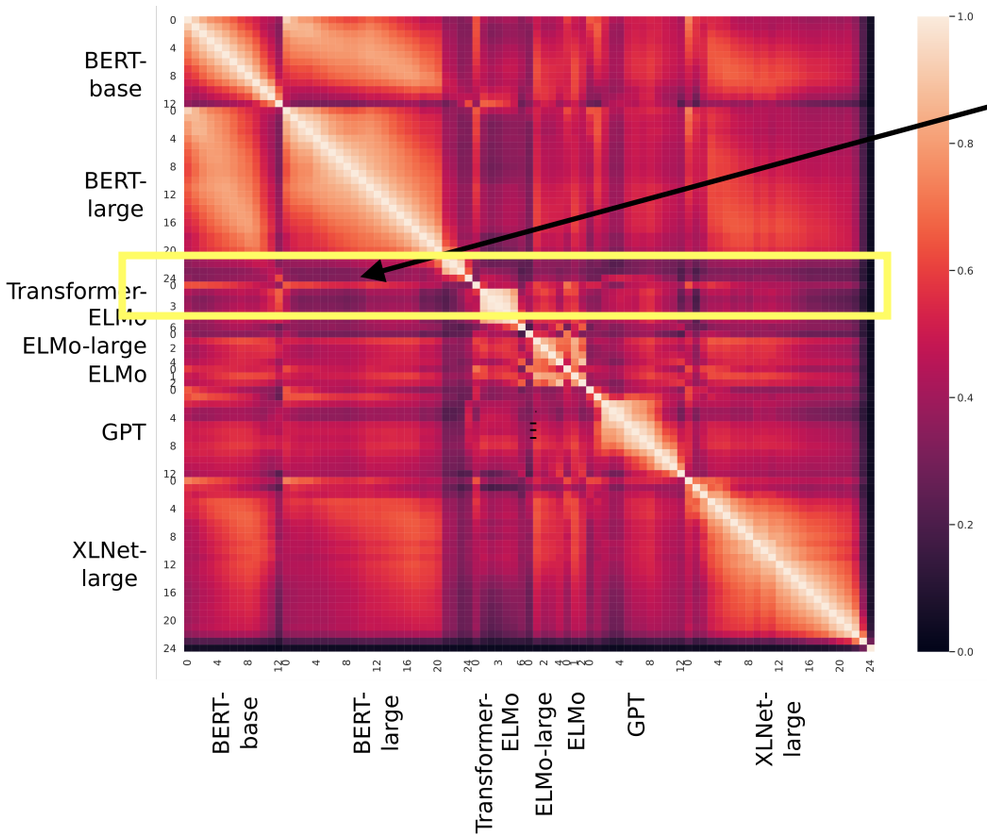


Figure 2: `ckasim` similarity heatmap of layers in base and random models.

- Comparing layers from pre-trained and randomly initialised models
- Still high intra-model similarities
- Random vs. pre-trained models are not very similar (as expected)
- Vertical bands in lower triangle
- Lower layers converge faster, leaving them closer to their initial random state.

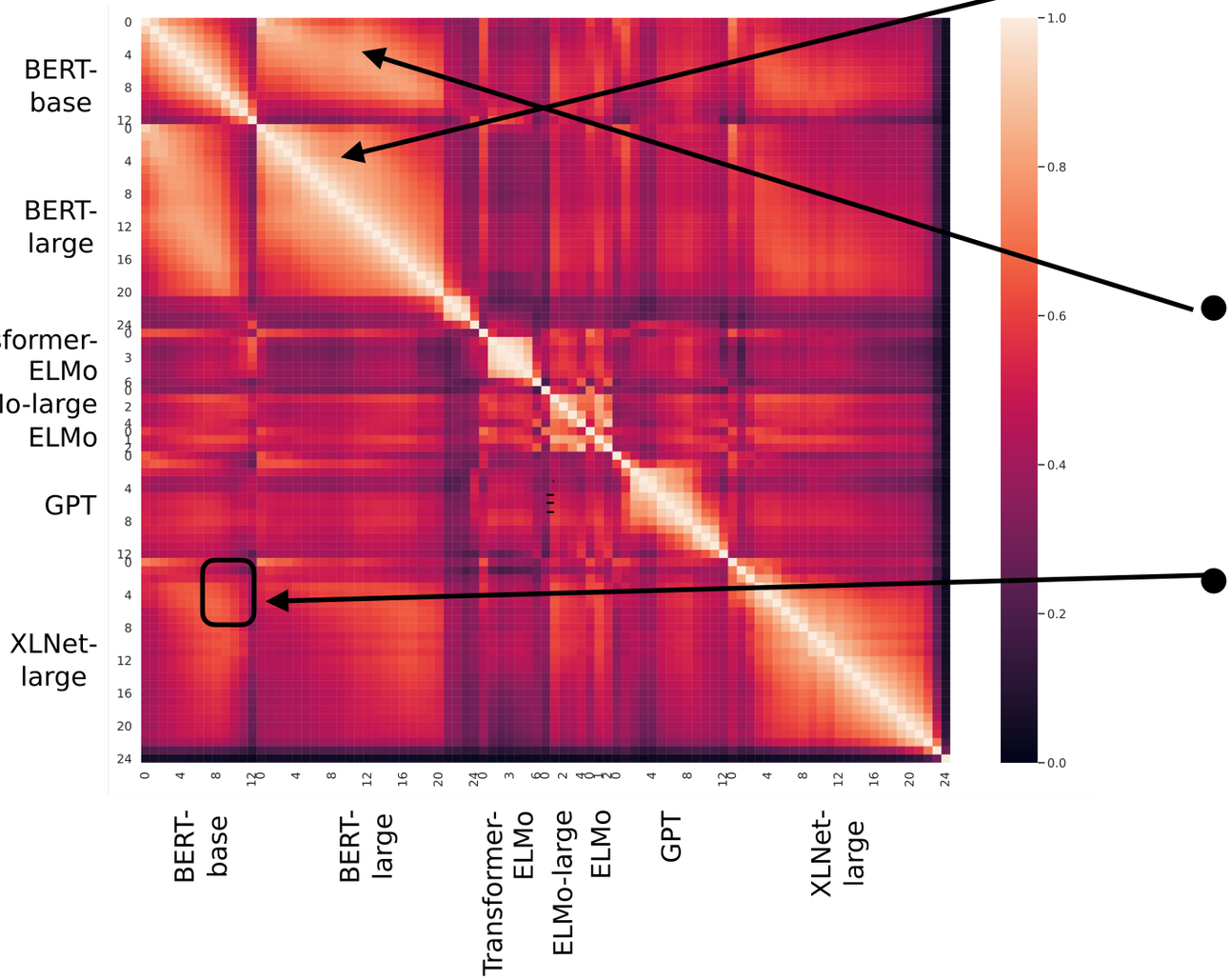
Lower layers



(b) `ckasim`

- Lower layers are more similar across architectures
  - Lower layers are closer to the input, which is always the same words

Adjacent layers are more similar



(b) `ckasim`

- There are “thick” bright diagonals, meaning that adjacent layers are more similar.
- BERT-base is similar to BERT-large as well
- Lower layers of XLNet are more similar to higher layers of BERT
- BERT requires mostly features from higher layers to achieve SoTA, when XLNet need lower and middle layers (Liu+19)

Higher layers are more localised than lower ones

- Average the neuronsim (local) and svssim (distributed) similarities between a layer and all the other layers. Subtract avg. neuronsim from avg. svssim to compute a “localisation score”

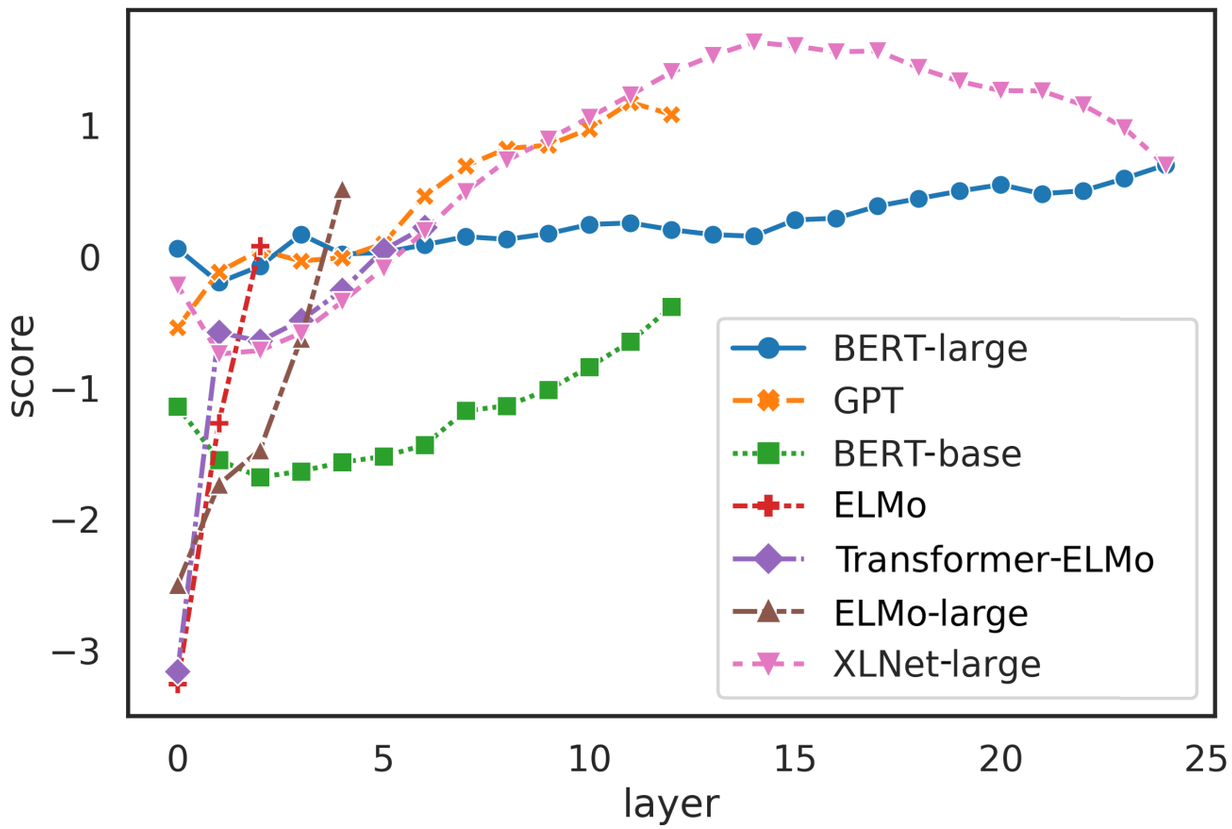
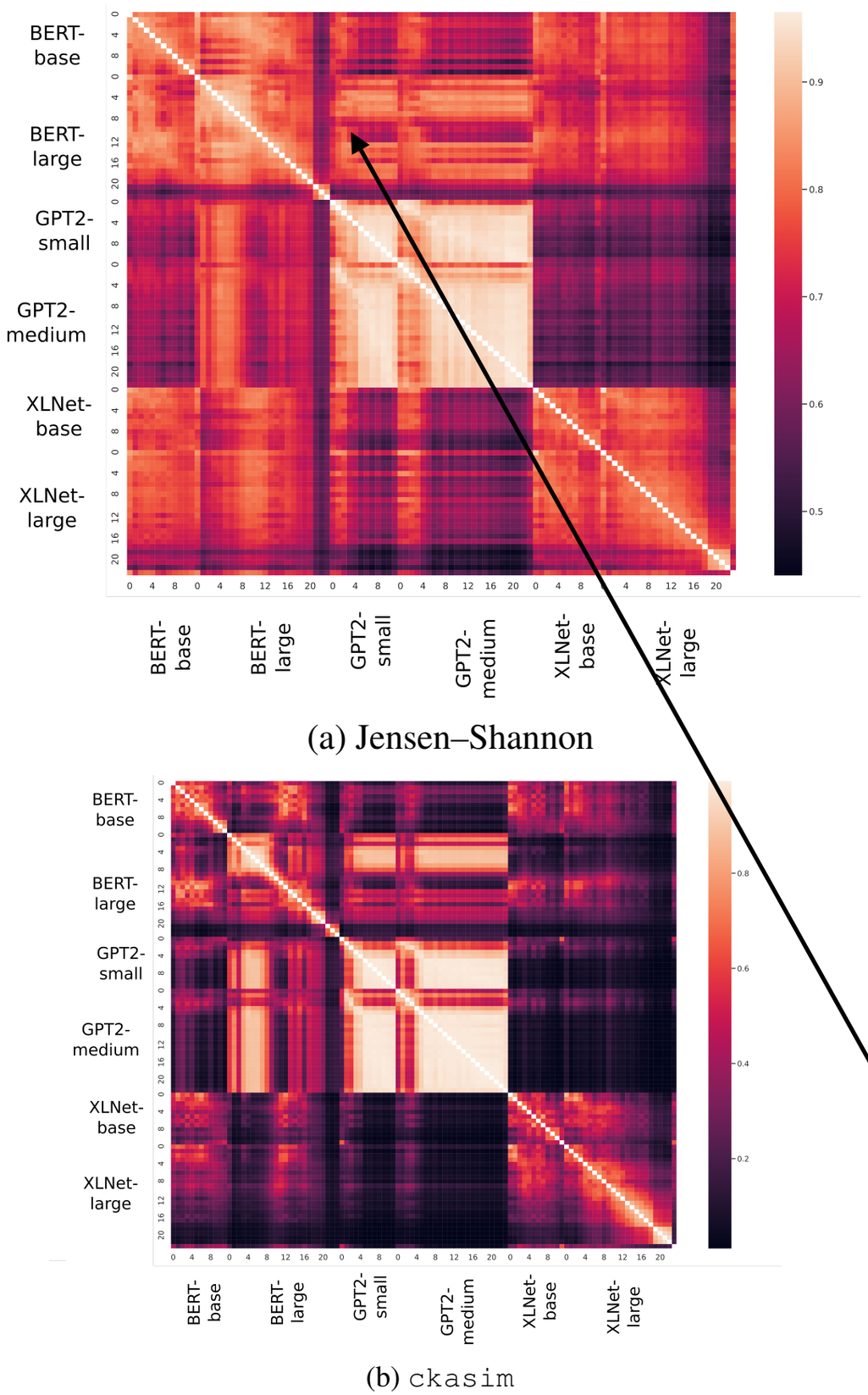


Figure 3: Localization score of various model layers.

- In all models, localisation score increases with layers
- Upper layers are more context-specific (Ethyarajah+19) — Same word’s embeddings are less similar in different contexts
- localised ≈ context-specific
- XLNet’s top layers are less context-specific

Attention Level



(a) Jensen–Shannon

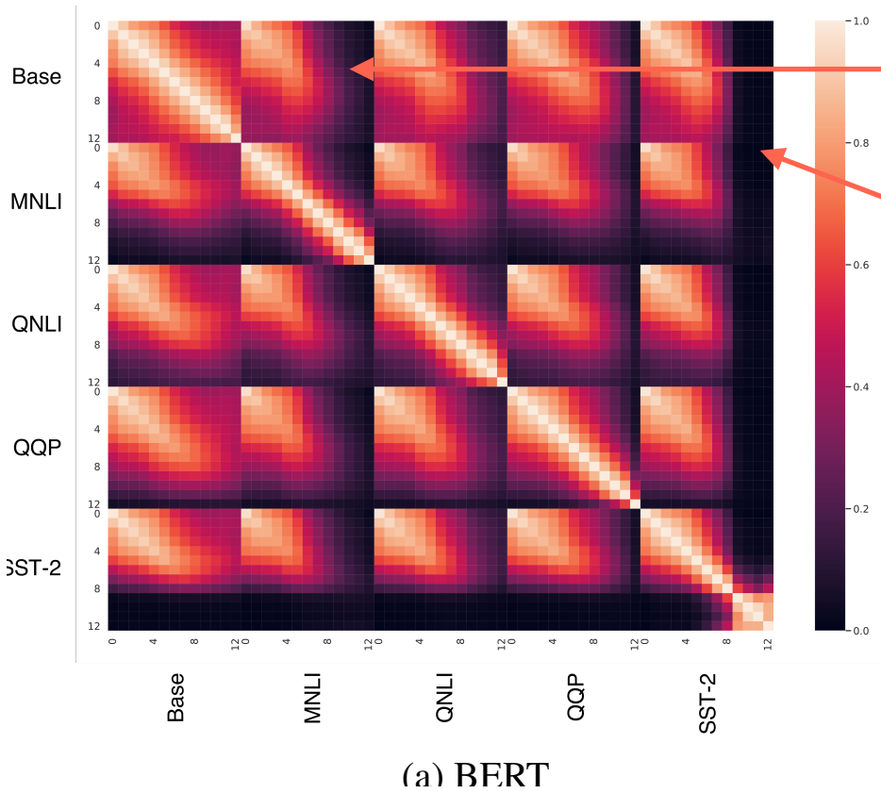
(b) `ckasim`

- Layers within the same model or model family are highly similar
- GPT2 layers are all similar to each other
- JS shows higher non-diagonal similarities than ckasim
- Difficult to identify patterns within a given model family (attention is difficult to interpret)
- GPT2 layers are similar to the bottom layers of BERT-large

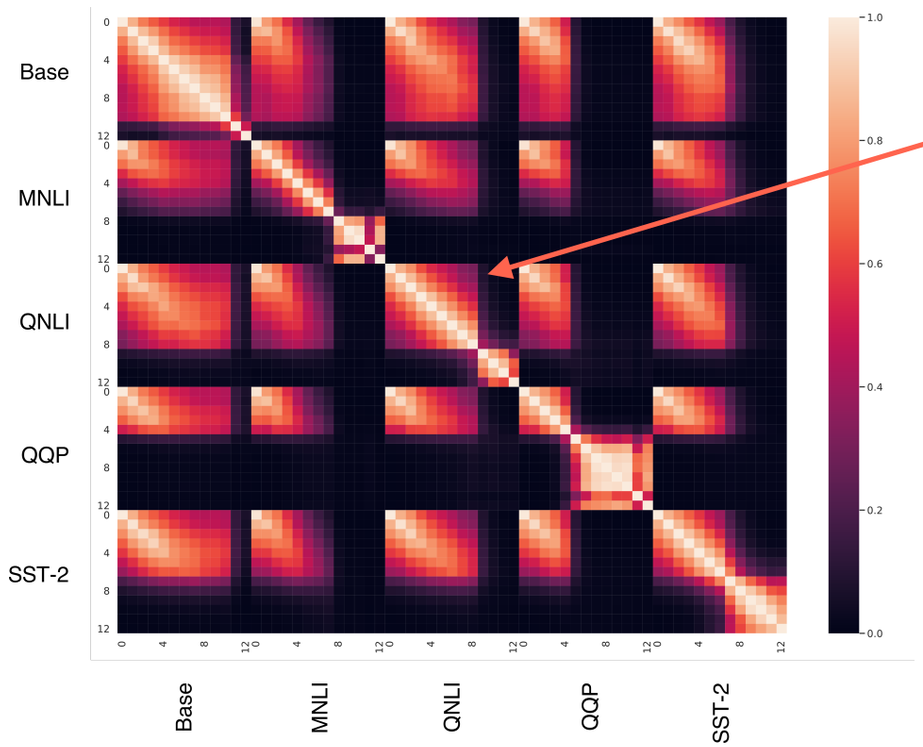
Similarity of Fine-tuned Models

- How does fine-tuning on downstream tasks affect model similarity?
  - Tasks:
    - MNLI (multilingual NLI)
    - QNLI (predict whether a sentence contains the answer to a question)
    - QQP: Classify whether two (Quora) questions are semantically equivalent (duplicate detection)
    - SST-2: binary sentiment classification on Stanford sentiment treebank

Similarity of Fine-tuned Models

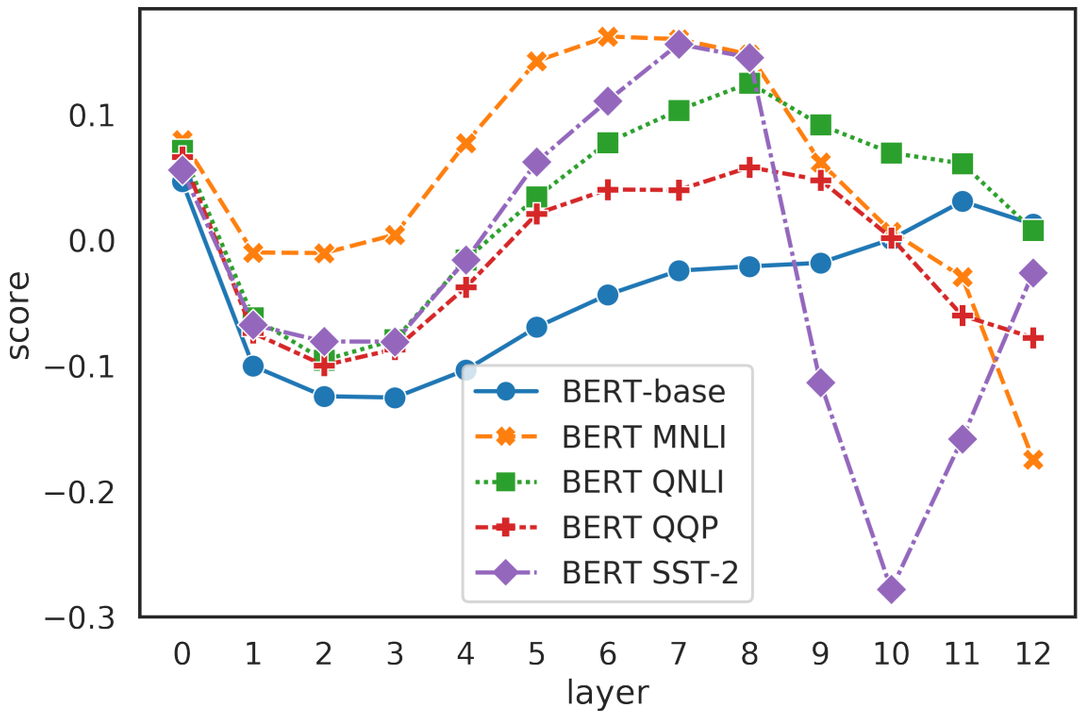


(a) BERT

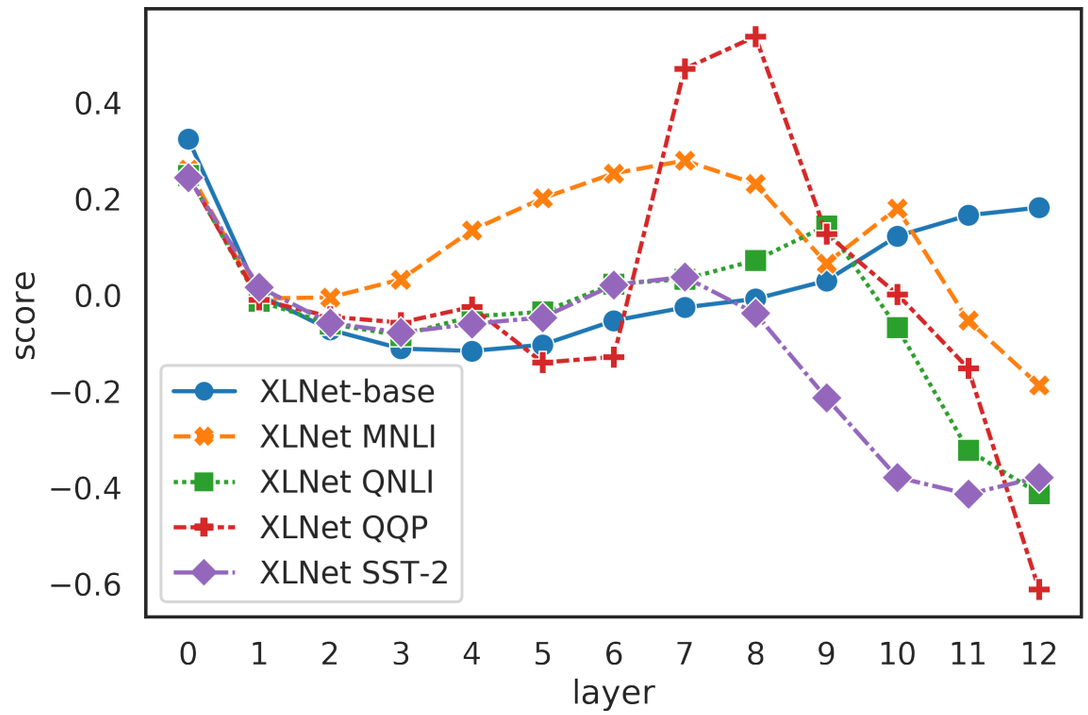


- Top layers are more affected (becomes less similar) after fine-tuning
- In BERT, top layers of SST-2 fine-tuned model are affected more
  - SST-2 is sentence classification whereas other tasks are sentence-pair classification
- In XLNet, fine-tuning on any task leads to top layers being very different from all layers of models fine-tuned on other tasks
- Difficult to multitask with XLNet compared to BERT?

Fine-tuning affects localisation



(a) BERT



(b) XLNet

- Compared to the pre-trained models, the fine-tuned ones decrease in localisation at the top layers.
  - To learn high-level tasks, we might require multiple specialised neurones.

Future directions

- Combine probing with similarity analysis (this paper) to identify which linguistic properties are captured in the model components that are similar to one another.

- Relationship between localisation of information and learnability of particular properties

- Context specificity vs. localisation connection

- Attention similarity vs. Interpretability/Explainability

- Even if two models have similar layers, the outputs could be very different because how those layers are connected (recurrent vs. transformer)