A. Reproducing Experiments and Figures

In this section, we present training and optimization details needed to reproduce our empirical validation of Theorem 1. We also published notebooks and check-pointed weights for two crucial experiments that investigate the result in the small and massive scale regimes, for Figure 1 and GPT-2 (https://github.com/google-research/google-research/tree/master/linear_identifiability).

A.1. Figure 1

We provide a Jupyter notebook and model checkpoints for reproducing Figure 1. Please refer to this for hyperparameter settings. In short, we implemented a model (Mnih & Teh, 2012) in the family of Section 2 and trained it on the Billion Word dataset (Chelba et al., 2013). This is illustrative of the property of Theorem 1 because the relatively modest size of the parameter space (see notebook) and massive dataset minimizes model convergence and data availability restrictions, e.g., approaches the asymptotic regime.

The word embedding space is 2-D for ease of visualization. We randomly selected a subset of words, mapped them into their learned embeddings, and visualized them as points in the left and middle panes. We then regress the points in pane one onto pane two in order to learn the optimal linear transformation between them. Note that if the two are linear transformations of each other, regression will recover that transformation exactly.

A.2. Simulation Study: Classification by DNNs

For this experiment, we want to ensure that the chosen model can fit the data distribution exactly. Controlling this removes one possible factor that could prevent linear identifiability of learned representations despite the model formally having that property. We do this by making sure that the process that generates the dataset matches the model chosen to learn the relationships between inputs and labels.

This is achieved through the following algorithm. We first randomly assign initialization labels based on angular position, then fit two neural networks $f_{\boldsymbol{\theta}^\star}$ and $g_{\boldsymbol{\theta}^\star}$ to predict the final labels, using the discriminative model of Equation (1) and Appendix D.1. Both $f_{\boldsymbol{\theta}^\star}$ and $g_{\boldsymbol{\theta}^\star}$ 4-hidden-layer MLPs with two 64 unit layers and one 2-D bottle neck layer. After training these representation functions to convergence, generated new batch of points $\mathbf{x}$, and used the trained networks to predict the ground truth labels $\mathbf{y}$.

Finally, to conduct experiments, we chose $\mathbf{f}_{\boldsymbol{\theta}'}$ and $\mathbf{g}_{\boldsymbol{\theta}'}$ to be the same architecture as $\mathbf{f}_{\boldsymbol{\theta}^\star}$ and $\mathbf{g}_{\boldsymbol{\theta}^\star}$. This ensures that the supervised classifier we attempted to learn would using the function approximators $\mathbf{f}_{\boldsymbol{\theta}'}$ and $\mathbf{g}_{\boldsymbol{\theta}'}$ would be able to capture the true data generating process, e.g, would not fail due to too few hidden units, or too complex a relationship between targets and inputs.

Remaining training details are as follows. We optimize weights using Adam with a learning rate of 10^{-4} for $5 * 10^4$ iterations. To make the classification problem more challenging, we additionally add 20 input dimensions of random noise to the data. The Adam optimizer (Kingma & Ba, 2014) with a learning rate of $3 \cdot 10^{-4}$ is used.

A.3. Self-Supervised Learning for Image Classification

To compute linear similarity between representations, we train two independent models in parallel. For each model we define both $\mathbf{f}_{\boldsymbol{\theta}}$ and $\mathbf{g}_{\boldsymbol{\theta}}$ as a 3-layer fully connected neural network with 2^8 units per layer and a fixed output dimensionality of 2^6. We define our model following Equation (1), where S is the set of the other image patches from the current minibatch and optimize the objective of (Hénaff et al., 2019). We augment both sampled patches independently with randomized brightness, saturation, hue, and contrast adjustments, following the recipe of (Hénaff et al., 2019). We train on the CIFAR10 dataset (Krizhevsky et al., 2009) with batchsize 2^8, using the Adam optimizer with a learning rate of 10^{-4} and the JAX (Bradbury et al., 2018) software package. For each model, we early stop based on a validation loss failing to improve further.

Additional details about the experiments that generated Figure 3:

Figure 3 a. Patches are sampled randomly from training images.

Figure 3 b. For each model, we train for at most $3 * 10^4$ iterations, early stopping when necessary based on validation loss.

Figure 3 c. For each model, we train for at most $3 * 10^4$ iterations, early stopping when necessary based on validation loss.

Figure 3 d. Error bars show standard error computed over 5 pairs of models after $1.5 * 10^4$ training iterations.

A.4. GPT-2

We include all details through a notebook in the code release. Pretrained GPT-2 weights as specified in the main text are publicly available from HuggingFace (Wolf et al., 2019).

A.5. Remark on Effect of Initialization and Hyperparameters of Models

One question that may be of interest is whether initialization affects whether learned representations will be within a linear transformation of each other. This depends on whether the optimization routines (like Adam, AdaGrad, etc.) are robust to wider initialization within a certain range. If so, model convergence will be unaffected. However, this cannot make up for poor initialization or poor optimization: just as in any deep neural network, a poor initialization and inadequate optimizer will interfere with learning the model parameters. In the case of a linearly identifiable model, means that the learned representations would not live within a linear transformation of each other (up to noise from model fitting), since the models have failed to converge to a reasonable solution for the task at hand.

When the hyperparameters of a DNN are changed, this changes the class of functions that the network can represent (i.e., the size and stride of convolution filters will change which input pixels could be correlated in deeper layers). Typically, hyperparameters are carefully tuned using cross validation based on held-out data. We did so in our experiments also. We expect that such a tuning procedure would yield hyperparameters that are as good as possible for the model to be optimized, allowing sufficient optimization so that the linear identifiability of the learned representations is realized. If the hyperparameters are sufficiently bad and optimization suffers, this will interfere with model fitting, and with linear identifiability of the learned representations also.

B. Proof that Linear Similarity is an Equivalence Relation

We claim that $\overset{\text{L}}{\sim}$ is an equivalence relation. It suffices to show that it is reflexive, transitive, and symmetric.

Proof. Consider some function $\mathbf{g}_{\boldsymbol{\theta}}$ and some $\boldsymbol{\theta}', \boldsymbol{\theta}^\star, \boldsymbol{\theta}^\dagger \subset \Theta$. Suppose $\boldsymbol{\theta}' \overset{\text{L}}{\sim} \boldsymbol{\theta}^\star$. Then, there exists an invertible matrix $\mathbf{B}$ such that $\mathbf{g}_{\boldsymbol{\theta}'}(\mathbf{x}) = \mathbf{B}\mathbf{g}_{\boldsymbol{\theta}^\star}(\mathbf{x})$. Since $\mathbf{g}_{\boldsymbol{\theta}^\star}(\mathbf{x}) = \mathbf{B}^{-1}\mathbf{g}_{\boldsymbol{\theta}'}(\mathbf{x})$, $\overset{\text{L}}{\sim}$ is symmetric. Reflexivity follows from setting $\mathbf{g}_{\boldsymbol{\theta}^\star}$ to $\mathbf{g}_{\boldsymbol{\theta}'}$ and $\mathbf{B}$ to the identity matrix. To show transitivity, suppose also that $\boldsymbol{\theta}^\star \overset{\text{L}}{\sim} \boldsymbol{\theta}^\dagger$. Then, there exists an invertible $\mathbf{C}$ such that $\mathbf{g}_{\boldsymbol{\theta}^\star}(\mathbf{x}) = \mathbf{C}\mathbf{g}_{\boldsymbol{\theta}^\dagger}(\mathbf{x})$. Since $\mathbf{g}_{\boldsymbol{\theta}'} \overset{\text{L}}{\sim} \mathbf{g}_{\boldsymbol{\theta}^\star}$, $\mathbf{B}^{-1}\mathbf{g}_{\boldsymbol{\theta}'}(\mathbf{x}) = \mathbf{C}\mathbf{g}_{\boldsymbol{\theta}^\dagger}(\mathbf{x})$. Rearranging terms, $\mathbf{g}_{\boldsymbol{\theta}'}(\mathbf{x}) = \mathbf{B}\mathbf{C}\mathbf{g}_{\boldsymbol{\theta}^\dagger}(\mathbf{x})$, so that $\boldsymbol{\theta}' \overset{\text{L}}{\sim} \boldsymbol{\theta}^\dagger$ as required. □

C. Section 3.2 Continued: Case of Context Representation Function g

Our derivation of identifiability of $\mathbf{g}_{\boldsymbol{\theta}}$ is similar to the derivation of $\mathbf{f}_{\boldsymbol{\theta}}$. The primary difference is that the normalizing constants in Equation (5) do not cancel out. First, note that we can rewrite Equation (1) as:

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}, \mathbf{S}) = \exp(\widetilde{\mathbf{f}}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{S})^\top \widetilde{\mathbf{g}}_{\boldsymbol{\theta}}(\mathbf{y})) \tag{8}$$

where:

$$\widetilde{\mathbf{f}}_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{S}) = [-Z(\mathbf{x}, \mathbf{S}); \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})] \tag{9}$$

$$\widetilde{\mathbf{g}}_{\boldsymbol{\theta}}(\mathbf{y}) = [1; \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y})] \tag{10}$$

$$Z(\mathbf{x}, \mathbf{S}) = \log \sum_{\mathbf{y}' \in \mathbf{S}} \exp(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^\top \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}')). \tag{11}$$

Below, we will show that for the model family defined in Section 2,

$$p_{\boldsymbol{\theta}'} = p_{\boldsymbol{\theta}^*} \implies \mathbf{g}_{\boldsymbol{\theta}'}(\mathbf{y}) = \mathbf{B}\,\mathbf{g}_{\boldsymbol{\theta}^\star}(\mathbf{y}), \tag{12}$$

where $\mathbf{B}$ is an invertible $(M \times M)$-dimensional matrix, concluding the proof of the linear identifiability of models in the family defined by Equation (1). We adopt the same shorthands as in the main text.

C.1. Diversity condition

We assume that for any $(\boldsymbol{\theta}', \boldsymbol{\theta}^*) \subset \Theta$ for which it holds that $p' = p^*$, and for any given $\mathbf{y}$, there exist $M+1$ tuples $\{(\mathbf{x}^{(i)}, \mathbf{S}^{(i)})\}_{i=0}^{M}$, such that $p_{\mathcal{D}}(\mathbf{x}^{(i)}, \mathbf{y}, \mathbf{S}^{(i)}) > 0$, and such that the $((M+1) \times (M+1))$ matrices $\mathbf{M}'$ and $\mathbf{M}^*$ are invertible, where $\mathbf{M}'$ consists of columns $\widetilde{\mathbf{f}}'(\mathbf{x}^{(i)}, \mathbf{S}^{(i)})$, and $\mathbf{M}^*$ consists of columns $\widetilde{\mathbf{f}}^*(\mathbf{x}^{(i)}, \mathbf{S}^{(i)})$.

This is similar to the diversity condition of Section 3.2 but milder, since a typical dataset will have multiple $\mathbf{x}$ for each $\mathbf{y}$.

C.2. Proof

With the data distribution $p_{\mathcal{D}}(\mathbf{x}, \mathbf{y}, \mathbf{S})$, for a given $\mathbf{y}$, there exists a conditional distribution $p_{\mathcal{D}}(\mathbf{x}, \mathbf{S}|\mathbf{y})$. Let $(\mathbf{x}, \mathbf{S})$ be a sample from this distribution. From Equation (1) and the statement to prove, it follows that:

$$p'(\mathbf{y}|\mathbf{x}, \mathbf{S}) = p^*(\mathbf{y}|\mathbf{x}, \mathbf{S}) \tag{13}$$

Substituting in the definition of our model from Equation (8), we find:

$$\exp(\widetilde{\mathbf{f}}'(\mathbf{x}, \mathbf{S})^\top \widetilde{\mathbf{g}}'(\mathbf{y})) = \exp(\widetilde{\mathbf{f}}^*(\mathbf{x}, \mathbf{S})^\top \widetilde{\mathbf{g}}^*(\mathbf{y})), \tag{14}$$

which, evaluating logarithms, becomes

$$\widetilde{\mathbf{f}}'(\mathbf{x}, \mathbf{S})^\top \widetilde{\mathbf{g}}'(\mathbf{y}) = \widetilde{\mathbf{f}}^*(\mathbf{x}, \mathbf{S})^\top \widetilde{\mathbf{g}}^*(\mathbf{y}), \tag{15}$$

which is true for any triple $(\mathbf{x}, \mathbf{y}, \mathbf{S})$ where $p_{\mathcal{D}}(\mathbf{y}|\mathbf{x}, \mathbf{S}) > 0$.

From $\mathbf{M}'$ and $\mathbf{M}^*$ (Section C.1) and Equation (15) we form a linear system of equations, collecting the $M+1$ relationships together:

$$\mathbf{M}'^{\top} \widetilde{\mathbf{g}}'(\mathbf{y}) = \mathbf{M}^{*\top} \widetilde{\mathbf{g}}^*(\mathbf{y}) \tag{16}$$

$$\widetilde{\mathbf{g}}'(\mathbf{y}) = \mathbf{A} \widetilde{\mathbf{g}}^*(\mathbf{y}), \tag{17}$$

where $\mathbf{A} = (\mathbf{M}^* \mathbf{M}'^{-1})^{\top}$, an invertible $(M+1) \times (M+1)$ matrix.

It remains to show the existence of an invertible $M \times M$ matrix $\mathbf{B}$ such that

$$\mathbf{g}'(\mathbf{y}) = \mathbf{B} \mathbf{g}^*(\mathbf{y}). \tag{18}$$

We proceed by constructing $\mathbf{B}$ from $\mathbf{A}$. Since $\mathbf{A}$ is invertible, there exist j elementary matrices $\{\mathbf{E}_1, \ldots, \mathbf{E}_j\}$ such that their action $\mathbf{R} = \mathbf{E}_j \mathbf{E}_{j-1} \ldots \mathbf{E}_1$ converts $\mathbf{A}$ to a (non-unique) row echelon form. Without loss of generality, we build $\mathbf{R}$ such that the $a_{1,1}$ entry of $\mathbf{A}$ is the first pivot, leading to the particular row echelon form:

$$\mathbf{RA} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \ldots & a_{1,m\times 1} \\ 0 & \tilde{a}_{2,2} & \tilde{a}_{2,3} & \ldots & \tilde{a}_{2,m\times 1} \\ 0 & 0 & \tilde{a}_{3,3} & \ldots & \tilde{a}_{2,m\times 1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & \tilde{a}_{m\times 1, m\times 1} \end{bmatrix}, \tag{19}$$

where $\tilde{a}_{i,j}$ indicates that the corresponding entry in $\mathbf{RA}$ may differ from $\mathbf{A}$ due to the action of $\mathbf{R}$. Applying $\mathbf{R}$ to Equation (16), we have

$$\mathbf{R} \widetilde{\mathbf{g}}'(\mathbf{y}) = \mathbf{RA} \widetilde{\mathbf{g}}^*(\mathbf{y}). \tag{20}$$

We now show that removing the first row and column of $\mathbf{RA}$ and $\mathbf{R}$ generates matrices of rank M. Let $\overline{\mathbf{RA}}$ and $\overline{\mathbf{R}}$ denote the $(M \times M)$ submatrices formed by removing the first row and column of $\mathbf{RA}$ and $\mathbf{R}$ respectively.

Equation (19) shows that $\overline{\mathbf{RA}}$ has a pivot in each column, and thus has rank M. To show that $\overline{\mathbf{R}}$ is invertible, we must show that removing the first row and column reduces the rank of $\mathbf{R} = \mathbf{E}_j \mathbf{E}_{j-1} \ldots \mathbf{E}_1$ by exactly 1. Clearly, each $\mathbf{E}_k$ is invertible, and their composition is invertible. We must show the same for the composition of $\overline{\mathbf{E}_k}$.

There are three cases to consider, corresponding to the three unique types of elementary matrices. Each elementary matrix acts on $\mathbf{A}$ by either (1) swapping rows i and j, (2) replacing row j by a multiple m of itself, or (3) adding a multiple m of row i to row j. We denote elementary matrix types by superscripts.

In Case (1), $\mathbf{E}_k^1$ is an identity matrix with row i and row j swapped. For Case (2), $\mathbf{E}_l^2$ is an identity matrix with the j, j^{th} entry replaced by some m. For each $\mathbf{E}_k^1$ and $\mathbf{E}_l^2$ in $\mathbf{R}$, where $1 \leq k, l \leq j$, we know that the indices $i, j \geq 2$, because we chose the first entry of the first row of $\mathbf{A}$ to be the pivot, and hence do not swap the first row, or replace the first row by itself multiplied by a constant. This implies that removing the first row and column of $\mathbf{E}_k^1$ and $\mathbf{E}_l^2$ removes a pivot entry 1 in the $(1,1)$ position, and removes zeros elsewhere. Hence, the $(M \times M)$ submatrices $\overline{\mathbf{E}_k^1}$ and $\overline{\mathbf{E}_l^2}$ are elementary matrices with rank M.

For Case (3), $\mathbf{E}_k^3$ has some value $m \in \mathbb{R}$ in the j, i^{th} entry, and 1s along the diagonal. In this case, we may find a non-zero entry in some $\mathbf{E}_k^3$, so that, e.g., the second row has a pivot at position $(2,2)$. Without loss of generality, suppose $i = 1$, $j = 2$ and let m be some nonzero constant. Removing the first row and column of $\mathbf{E}_1^3$ removes this m also. Nevertheless, $\overline{\mathbf{E}_1^3} = \mathbf{I}_M$, the rank M identity matrix. For any other $\mathbf{E}_k^3$ $1 < i \leq M+1$, $j \geq 2$ because we chose $a_{1,1}$ as the first pivot, and hence do not swap the first row, or replace the first row by itself multiplied by a constant. In both cases, removing the first row and first column creates an $\overline{\mathbf{E}_k^3}$ that is a rank M elementary matrix.

We have shown by the above that $\overline{\mathbf{R}}$ is a composition of rank M matrices. We now construct the matrix $\mathbf{B}$ by removing the first entries of $\widetilde{\mathbf{g}}'$ and $\widetilde{\mathbf{g}}^\star$, and removing the first row and first column of $\mathbf{R}$ and $\mathbf{RA}$ in Equation (20). Then, we have

$$\overline{\mathbf{R}} \mathbf{g}'(\mathbf{y}) = \overline{\mathbf{RA}} \mathbf{g}^*(\mathbf{y}), \tag{21}$$

$$\mathbf{g}'(\mathbf{y}) = \overline{\mathbf{R}}^{-1} \overline{\mathbf{RA}} \mathbf{g}^*(\mathbf{y}). \tag{22}$$

Choosing $\mathbf{B} = \overline{\mathbf{R}}^{-1} \overline{\mathbf{RA}}$ proves the result.

□

D. Reductions to Canonical Form of Eq. (1)

In the following, we show membership in the model family of Equation (1) using the mathematical notation of the papers under discussion in Section 4. Note that each subsection will change notation to match the papers under discussion, which varies quite widely. We employ the following colour-coding scheme to aid in clarity:

$$\log p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}, \mathbf{S}) = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^{\top} \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}) - \log \sum_{\mathbf{y}' \in \mathbf{S}} \exp(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^{\top} \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}')),$$

where: $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})$ is generalized to a data representation function, $\mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y})$ is generalized to a context representation function, and $\sum_{\mathbf{y}' \in \mathbf{S}} \exp(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^{\top} \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{y}'))$ is a constant.

D.1. Supervised Classification

Supervised classifiers commonly employ a neural network feature extractor followed by a linear projection of the out-

put of this network into a space of unnormalized logits. All the layers prior to the logits are the representation function $\mathbf{f}_{\boldsymbol{\theta}}$, and the final projection layer is the context map $\mathbf{g}_{\boldsymbol{\theta}}(y = i) = \mathbf{w}_i$, where $\mathbf{w}_i$ is the i-th column of a weight matrix $\mathbf{W}$. The set $\mathbf{S}$ in this case contains human-chosen labels and has no stochasticity. The loss function is the negative log-likelihood of the data under a categorical distribution with a softmax parameterization:

$$\log p_{\boldsymbol{\theta}}(y = i|\mathbf{x}; \mathbf{S}) = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^\top \boldsymbol{w}_i - \log \sum_{j=1}^{|\mathbf{S}|} \exp(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})^\top \boldsymbol{w}_j)$$

Supervised classification is thus an member of the family defined in Section 2. It exhibits the simplest functional form for the $\mathbf{g}$ function while allowing $\mathbf{f}$ to be arbitrarily complicated.

D.2. Contrastive Predictive Coding (CPC)

Consider a sequence of points $\mathbf{x}_t$. We wish to learn the parameters $\boldsymbol{\phi}$ to maximize the k-step ahead predictive distribution $p(\mathbf{x}_{t+k}|\mathbf{x}_t, \boldsymbol{\phi})$. In the image patch example, each patch center i, j is indexed by t. Each $\mathbf{x}_t$ is mapped to a sequence of feature vectors $\boldsymbol{z}_t = f_\theta(\mathbf{x}_t)$ An autoregressive model, already updated with the previous latent representations $\boldsymbol{z}_{\leq t-1}$, transforms the $\boldsymbol{z}_t$ into a "context" latent representation $\mathbf{c}_t = g_{AR}(\boldsymbol{z}_{\leq t})$. Instead of predicting future observations k steps ahead, $\mathbf{x}_{t+k}$, directly through a generative model $p_k(\mathbf{x}_{t+k}|\mathbf{c}_t)$, Oord et al. (2018) model a density ratio in order to preserve the mutual information between $\mathbf{x}_{t+k}$ and $\mathbf{c}_t$.

Objective Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ be a set of N random samples containing one positive sample from $p(\mathbf{x}_{t+k}|\mathbf{c}_t)$ and $N - 1$ samples from the proposal distribution $p(\mathbf{x}_{t+k})$. Oord et al. (2018) define the following link function: $l_k(\mathbf{x}_{t+k}, \mathbf{c}_t) \triangleq \exp\left(\mathbf{z}_{t+k}^\top \mathbf{W}_k \mathbf{c}_t\right)$. Then, CPC optimizes

$$-\mathbb{E}_{\mathbf{X}}\left[\log \frac{l_k(\mathbf{x}_{t+k}, \mathbf{c}_t)}{\sum_{x_j \in X} l_k(\mathbf{x}_j, \mathbf{c}_t)}\right] \tag{23}$$

$$= -\mathbb{E}_{\mathbf{X}}\left[\log \frac{\exp\left(\mathbf{z}_{t+k}^\top \mathbf{W}_k \mathbf{c}_t\right)}{\sum_{\mathbf{x}_j \in \mathbf{X}} \exp\left(\mathbf{z}_j^\top \mathbf{W}_k \mathbf{c}_t\right)}\right]. \tag{24}$$

Substituting in the definition of l_k makes Equation (24) identical to the model family Equation (1).

D.3. Autoregressive language models (e.g. GPT-2)

Let $\mathcal{U} = \{u_1, \ldots, u_n\}$ be a corpus of tokens. Autoregressive language models maximize a log-likelihood $L(\mathcal{U}) = \sum_{i=1}^n \log P(u_i|u_{i-k}, \ldots, u_{i-1}; \Theta)$, Concretely, the conditional density is modelled as

$$\log P(u_i|u_{i-k:i-1}; \Theta) = \mathbf{W}_i \mathbf{h}_i - \log \sum_j \exp(\mathbf{W}_j \mathbf{h}_i),$$

where $\mathbf{h}_i$ is the $m \times 1$ output of a function approximator (e.g. a Transformer decoder (Liu et al., 2018)), and $\mathbf{W}_{i:}$ is the i'th row of the $|\mathcal{U}| \times m$ token embedding matrix.

D.4. BERT

Consider a sequence of text $\mathbf{x} = [x_1, \ldots, x_T]$. Some proportion of the symbols in $\mathbf{x}$ are extracted into a vector $\bar{\mathbf{x}}$, and then set in $\mathbf{x}$ to a special null symbol, "corrupting" the original sequence. This operation generates the corrupted sequence $\underline{\mathbf{x}}$. The representational learning task is to predict $\bar{\mathbf{x}}$ conditioned on $\underline{\mathbf{x}}$, that is, to maximize w.r.t. $\boldsymbol{\theta}$:

$$\log p_\theta(\bar{\mathbf{x}}|\underline{\mathbf{x}}) \approx \sum_{t=1}^T m_t \log p_\theta(x_t|\underline{\mathbf{x}})$$

$$= \sum_{t=1}^T m_t \left(H_\theta(\underline{\mathbf{x}})_t^\top e(x_t) - \log \sum_{x'} \exp\left(H_\theta(\underline{\mathbf{x}})_t^\top e(x')\right) \right),$$

where H is a transformer, e is a lookup table, and $m_t = 1$ if symbol x_t is masked. That is, corrupted symbols are "reconstructed" by the model, meaning that their index is predicted. As noted in Yang et al. (2019), BERT models the joint conditional probability $p(\bar{\mathbf{x}}|\underline{\mathbf{x}})$ as factorized so that each masked token is separately reconstructed. This means that the log likelihood is approximate instead of exact.

D.5. QuickThought Vectors

Let $\mathbf{f}$ and $\mathbf{g}$ be functions that take a sentence as input and encode it into an fixed length vector. Let s be a given sentence, and S_{ctxt} be the set of sentences appearing in the context of s for a fixed context size. Let S_{cand} be the set of candidate sentences considered for a given context sentence $s_{ctxt} \in S_{ctxt}$. Then, S_{cand} contains a valid context sentence s_{ctxt} as well as many other non-context sentences. S_{cand} is used for the classification objective. For any given sentence position in the context of s (for example, the preceding sentence), the probability that a candidate sentence $s_{cand} \in S_{cand}$ is the correct sentence for that position is given by $\log p(s_{cand}|s, S_{cand})$ which equals

$$f_\theta(s)^\top g_\theta(s_{cand})) - \log \sum_{s' \in S_{cand}} \exp\left(f_\theta(s)^\top g_\theta(s'_{cand})\right).$$

D.6. Deep Metric Learning

The *multi-class N-pair loss* in Sohn (2016) is proportional to

$$\log N - \frac{1}{N}\sum_{i=1}^{N}\log\left(1+\sum_{j\neq i} e^{\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_j) - \mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_i)}\right),$$

which can be simplified as

$$\begin{aligned}
&-\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{K}\sum_{j=1}^{K} e^{\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_j) - \mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_i)}\right)\\
&=\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{1}{\frac{1}{K}\sum_{j=1}^{K} e^{\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_j) - \mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_i)}}\right)\\
&=\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{e^{\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_i)}}{\frac{1}{K}\sum_{j=1}^{K} e^{\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_j)}}\right).
\end{aligned}$$

Setting N to 1 and evaluating the log gives

$$\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_i) - \frac{1}{K}\sum_{j=1}^{K}\exp(\mathbf{f}_{\boldsymbol{\theta}}(x_i)^\top \mathbf{f}_{\boldsymbol{\theta}}(y_j)),$$

which is Equation (1) where $\mathbf{f}_{\boldsymbol{\theta}} = \mathbf{g}_{\boldsymbol{\theta}}$.

D.7. Neural Probabilistic Language Models (NPLMs)

Figure 1 shows results from a neural probabilistic language model as proposed in Mnih & Teh (2012). Mnih & Teh (2012) propose using a log-bilinear model (Mnih & Hinton, 2009) which, given some context h, learns a context word vectors r_w and target word vectors q_w. Two different embedding matrices are maintained, in other words: one to capture the embedding of the word and the other the context. The representation for the context vector, $\hat{q}$, is then computed as the linear combination of the context words and a context weight matrix C_i so that $\hat{q} = \sum_{i=1}^{n-1} C_i r_{w_i}$. The score for the match between the context and the next word is computed as a dot product, e.g., $s_\theta(w,h) = \hat{q}^\top \tilde{q}_w$[2] and substituting into the definition of $P_\theta^h(w)$, we see that

$$\log P_\theta^h(w) = \hat{q}^\top \tilde{q}_w - \log\sum_{w'}\exp\left(\hat{q}^\top \tilde{q}_{w'}\right)$$

shows that Mnih & Teh (2012) is a member of the model family.

Interestingly, a touchstone work in the area of NPLMs, Word2Vec (Mikolov et al., 2013), does not fall under the model family due to an additional nonlinearity applied to the score of Mnih & Teh (2012).

[2]We have absorbed the per-token baseline offset b into the q_w defined in Mnih & Teh (2012), forming the vector $\tilde{q}_w$ whose i'th entry is $(q_w)_i = (q_w)_i + b_w/(\hat{q})_i$

References

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., and Wanderman-milne, S. Jax: Composable transformations of Python+NumPy programs, 2018. URL Http://Github.Com/Google/Jax.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., and Others. Language Models are Few-Shot Learners. *Arxiv Preprint Arxiv:2005.14165*, 2020.

Chelba, C., Mikolov, T., Schuster, M., Ge, Q., Brants, T., Koehn, P., and Robinson, t. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *Arxiv Preprint Arxiv:1312.3005*, 2013.

Dai, A. M. and Le, Q. V. Semi-Supervised Sequence Learning. In *Advances in Neural information Processing Systems*, pp. 3079–3087, 2015.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Arxiv Preprint Arxiv:1810.04805*, 2018.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P.-A., Vincent, P., and Bengio, S. Why Does Unsupervised Pretraining Help Deep Learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Hénaff, O. J., Razavi, A., Doersch, C., Eslami, S., and Oord, A. V. D. Data-Efficient Image Recognition with Contrastive Predictive Coding. *Arxiv Preprint Arxiv:1905.09272*, 2019.

Hochreiter, S. and Schmidhuber, J. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

Hoffer, E. and Ailon, N. Deep Metric Learning Using Triplet Network. In *International Workshop On Similarity-Based Pattern Recognition*, pp. 84–92. Springer, 2015.

Hornik, K., Stinchcombe, M., and White, H. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2(5):359–366, 1989.

Hotelling, H. Relations Between Two Sets of Variates. *Biometrika*, 28(3/4):321–377, 1936.

Hyvärinen, A. and Morioka, H. Unsupervised Feature Extraction by Time-Contrastive Learning and Nonlinear ICA. In *Advances in Neural information Processing Systems*, pp. 3765–3773, 2016.

Hyvärinen, A., Sasaki, H., and Turner, R. E. Nonlinear ICA Using Auxiliary Variables and Generalized Contrastive Learning. *Arxiv Preprint Arxiv:1805.08651*, 2018.

Jacot, A., Gabriel, F., and Hongler, C. Neural Tangent Kernel: Convergence and Generalization in Neural Networks. In *Advances in Neural information Processing Systems*, pp. 8571–8580, 2018.

Johansson, F., Shalit, U., and Sontag, D. Learning representations for counterfactual inference. In *International conference on machine learning*, pp. 3020–3029, 2016.

Khemakhem, I., Kingma, D. P., and Hyvärinen, A. Variational Autoencoders and Nonlinear ICA: A Unifying Framework. *Arxiv Preprint Arxiv:1907.04809*, 2019.

Khemakhem, I., Monti, R. P., Kingma, D. P., and Hyvärinen, A. ICE-BeeM: Identifiable Conditional Energy-based Deep Models. *Arxiv Preprint Arxiv:2002.11537*, 2020.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. *Arxiv Preprint Arxiv:1412.6980*, 2014.

Krizhevsky, A., Hinton, G., and Others. Learning Multiple Layers of Features from Tiny Images. 2009.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-dickstein, J. Deep Neural Networks as Gaussian Processes. *Arxiv Preprint Arxiv:1711.00165*, 2017.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. Generating Wikipedia by Summarizing Long Sequences. *Arxiv Preprint Arxiv:1801.10198*, 2018.

Louizos, C., Shalit, U., Mooij, J. M., Sontag, D., Zemel, R., and Welling, M. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pp. 6446–6456, 2017.

Mikolov, T., Karafiát, M., Burget, L., Černockỳ, J., and Khudanpur, S. Recurrent Neural Network Based Language Model. In *Eleventh Annual Conference of The international Speech Communication Association*, 2010.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural information Processing Systems*, pp. 3111–3119, 2013.

Mnih, A. and Hinton, G. E. A Scalable Hierarchical Distributed Language Model. In *Advances in Neural information Processing Systems*, pp. 1081–1088, 2009.

Mnih, A. and Teh, Y. W. A Fast and Simple Algorithm for Training Neural Probabilistic Language Models. *Arxiv Preprint Arxiv:1206.6426*, 2012.

Morcos, A. S., Raghu, M., and Bengio, S. Insights on Representational Similarity in Neural Networks with Canonical Correlation, 2018.

Neal, R. M. *Bayesian Learning for Neural Networks*. PhD thesis, University of toronto, 1995.

Oord, A. V. D., Li, Y., and Vinyals, O. Representation Learning with Contrastive Predictive Coding. *Arxiv Preprint Arxiv:1807.03748*, 2018.

Pearson, K. LIII. On Lines and Planes of Closest Fit to Systems of Points in Space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving Language Understanding by Generative Pre-training. 2018.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language Models are Unsupervised Multitask Learners. *Openai Blog*, 1(8), 2019.

Raghu, M., Gilmer, J., Yosinski, J., and Sohl-Dickstein, J. SVCCA: Singular Vector Canonical Correlation Analysis for Deep Learning Dynamics and interpretability. In *Advances in Neural information Processing Systems*, pp. 6076–6085, 2017.

Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. CNN Features Off-the-Shelf: An Astounding Baseline for Recognition. In *Proceedings of The Ieee Conference On Computer Vision and Pattern Recognition Workshops*, pp. 806–813, 2014.

Sohn, K. Improved Deep Metric Learning with Multi-class N-Pair Loss Objective. In *Advances in Neural information Processing Systems*, pp. 1857–1865, 2016.

Sorrenson, P., Rother, C., and Köthe, U. Disentanglement by Nonlinear ICA with General Incompressible-flow Networks (Gin). *Arxiv:2001.04872 [Cs, Stat]*, January 2020.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is All You Need. In *Advances in Neural information Processing Systems*, pp. 5998–6008, 2017.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of The 25th international Conference On Machine Learning*, pp. 1096–1103, 2008.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. Huggingface's Transformers: State-of-the-art Natural Language Processing. *Arxiv*, Abs/1910.03771, 2019.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. XLNET: Generalized Autoregressive Pretraining for Language Understanding. *Arxiv Preprint Arxiv:1906.08237*, 2019.