By Tying Embeddings You Are Assuming the Distributional Hypothesis

Francesco Bertolotti [*1] **Walter Cazzola** [*1]

Abstract

In this work, we analyze both theoretically and empirically the effect of tied input-output embeddings—a popular technique that reduces the model size while often improving training. Interestingly, we found that this technique is connected to Harris (1954)'s distributional hypothesis—often portrayed by the famous Firth (1957)'s quote "*a word is characterized by the company it keeps*". Specifically, our findings indicate that words (or, more broadly, symbols) with similar semantics tend to be encoded in similar input embeddings, while words that appear in similar contexts are encoded in similar output embeddings (thus explaining the semantic space arising in input and output embedding of foundational language models). As a consequence of these findings, the tying of the input and output embeddings is encouraged only when the distributional hypothesis holds for the underlying data. These results also provide insight into the embeddings of foundation language models (which are known to be semantically organized). Further, we complement the theoretical findings with several experiments supporting the claims.

1. Introduction

Masked Language Modeling (MLM) (Devlin et al., 2019) and Causal Language Modeling (CLM) (Radford et al., 2019) have become one of the most influential training techniques for foundation models (Touvron et al., 2023; Gunasekar et al., 2023; Chowdhery et al., 2023; Raffel et al., 2020; Zhao et al., 2023) in the context of Natural Language Processing (NLP).

One of the core components of these models are embeddings—tables mapping words to trainable parameter vectors. It has been shown that foundation models, during training, organize word embeddings into a semantic space (Wu et al., 2020; Wang et al., 2020) similarly to word models such as Word2Vec (Mikolov et al., 2013). While word models have been extensively studied (Levy & Goldberg, 2014; Li et al., 2015; Yin & Shen, 2018), To the best of our knowledge, the process responsible for the emergence of a semantic structure, although well documented (Turian et al., 2010) lacks a formal treatment.

In this work, we aim to address this issue by introducing a concept already well-known in the context of programming languages—semantic equivalence. Briefly, two programs, p_1 and p_2, are semantically equivalent if they produce the same output when given the same initial state ρ ($\forall \rho : [\![p_1]\!](\rho) = [\![p_2]\!](\rho)$) (Scott & Strachey, 1971). From this definition, we can derive the notion of semantic equivalence for words, or more in general for symbols. Equipped with this notion, we will be able to study and formally investigate why foundation models organize their embeddings into a semantic space.

Remarkably, our findings indicate that under optimality and distributional assumptions, both input and output embeddings must encode the same semantic information. We believe that this result explains the effectiveness and popularity (Nguyen & Salazar, 2019; Levine et al., 2020; Cho et al., 2021) of weight tying (also known as shared input-output embeddings) (Inan et al., 2016; Press & Wolf, 2017).

Research Questions In this work, we aim to answer the following research questions:

- **RQ$_1$ Why the input embeddings in foundation models encode semantic information?**
- **RQ$_2$ Why the output embeddings in foundation models encode semantic information?**
- **RQ$_3$ Why input-output embeddings weight tying is effective?**

2. Background

Input Embeddings. The input embedding layer is the first layer found in language models. It maps symbols to vectors of trainable parameters. The embedding layer is usually represented with a parameter matrix $E^I \in \mathbb{R}^{V \times d}$. Where V is the vocabulary size and d is the size of each

[*]Equal contribution [1]Department of Computer Science, Università degli Studi di Milano, Milan, Italy. Correspondence to: Walter Cazzola <cazzola@di.unimi.it>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

embedding. Here, the embedding representing the j-th symbol corresponds to the j-th row of the matrix E^I—denoted $E^I(j)$.

Output Embeddings. The output embedding layer coincides with the last linear layer of language models. It maps the last encoding vector to the logit vector on the target vocabulary. In some cases, the last linear layer is substituted with an affine layer which includes the presence of a bias term. In this work, we assume that the last layer of the network is linear. In this case, the layer can be represented with a single parameter matrix $E^O \in \mathbb{R}^{d \times V}$. Here, the embedding representing the j-th symbol corresponds to the j-th column of the matrix E^O—denoted $E^O(j)$.

Weight Tying. Weight tying, in general, is the technique of sharing trainable parameters from different layers (as in (Lan et al., 2019)). Weight Tying has the beneficial effect of reducing the memory footprint of the network without heavily compromising performance. Surprisingly, the input-output embeddings weight tying (meaning that $E^I = (E^O)^T$) (Inan et al., 2016; Press & Wolf, 2017) is often found to lead to faster training and better generalizations (Pappas et al., 2018).

3. Semantic & Conditional Equivalence

Let us formalize the learning problem. Consider a dataset of input-output pairs $D = \{(x_i, y_i)\}_{i=1}^N$, where $y_i \in \mathcal{Y}$ represents the possible output and $x_i \in \mathcal{X}$ denotes the possible inputs. In the supervised and self-supervised setting, the objective is to learn a predictor mapping inputs to the respective outputs. Further, we define an input as a sequence of symbols from an alphabet Σ ($\mathcal{X} \subseteq \Sigma^*$) and an output as a single symbol from another alphabet Δ ($\mathcal{Y} = \Delta$). For example,

$$(x_i = \sigma_1\ \sigma_2\ \sigma_3\ \sigma_4\ \sigma_5\ \sigma_6,\ y_i = \delta_1)$$

denotes a possible sample from the dataset D.

Two popular techniques devised in the field of NLP are MLM and CLM. Here, the set of input symbols (Σ) corresponds to a vocabulary of tokens augmented with a few special tokens. We are concerned only with the *mask* token denoted with a question mark symbol ($? \in \Sigma$). The set of output symbols is equal to the set of input symbols ($\Sigma = \Delta$). Further, from each naturally occurring sentence, we replace one token with a mask token, and the replaced token becomes the output label. For example,

$$(x_i = \sigma_1\ \sigma_2\ \sigma_3\ ?\ \sigma_5\ \sigma_6,\ y_i = \sigma_4)$$

denotes a possible sample from a self-supervised dataset. In other words, the goal is to recover the masked symbol using only the surrounding symbols. If the masked symbol can appear only at the end of the input sequence, we speak of CLM, otherwise, we speak of MLM. In the context of NLP, these symbols usually take the form of words. However, similar modeling can be applied to other data modalities such as audio (Chen et al., 2020), time series (Rosin et al., 2022), and even images (He et al., 2022; Li et al., 2023a). For simplicity, we make the assumption that only a single token is masked in any given input. However, it is worth noting that in practice, many language models mask and retrieve several tokens simultaneously.

Given this context, a natural question arises: when two symbols can be regarded as having the same, or similar, meaning? To answer this question, we can turn to the notion of semantic equivalence in programming languages. Using denotational semantics, two programs, p_1 and p_2, are semantically equivalent iff $\forall \rho :\ [\![p_1]\!](\rho) = [\![p_2]\!](\rho)$. Meaning that both p_1 and p_2 produce the same output state when executed in the same initial state ρ. Now, let us replace the output state with a probability distribution over the alphabet and the programs with symbols. This yields the following definition:

Definition 3.1 (Semantically Equivalent Symbols)**.** Symbols $u, v \in \Sigma$ are said semantically equivalent (denoted $u \triangleq v$) iff

$$\forall y \in \Delta, \rho \in \mathrm{P} : p(y|\rho, u) = p(y|\rho, v)$$

Here, We will use the symbol ρ and P to denote the context of another symbol and the set of possible context respectively. On the left side, the symbol u is inside context ρ. On the right side, the symbol v is inside ρ. Intuitively, this means that replacing a symbol with a semantically equivalent one has no effect on the output distribution. Similarly, we can define the notion of semantically similar symbols:

Definition 3.2 (Semantically Similar Symbols)**.** Symbols $u, v \in \Sigma$ are said semantically similar iff

$$d([p(y|\rho, u)]_{y \in \mathcal{Y}, \rho \in \mathrm{P}},\ [p(y|\rho, v)]_{y \in \mathcal{Y}, \rho \in \mathrm{P}}) \leq \epsilon$$

Here, d denotes a distance measure (e.g. Euclidean distance). Intuitively, this means that a symbol can be replaced with a semantically similar one without heavily affecting the output distribution in most contexts.

In the context of NLP, semantically equivalent/similar symbols are, for example, synonyms and antonyms. Swapping a word with one of its synonyms should not have much of an impact on the distribution of masked tokens. To a lesser extent the principle holds also for antonyms. Consider the example:

`the ?  of water is half empty/full`

Regardless of using the word `empty` or `full` in this sentence, it will not change the outcome distribution of the masked symbol. In this case, we would see words such as `glass` or `cup` to be those with higher probability. If we could generalize this reasoning for any/most context, we would say that `empty` and `full` are semantically equivalent/similar.

Next, we focus on different kind of equivalence:

Definition 3.3 (Conditionally Equivalent Symbols)**.** Symbols $u, v \in \Sigma$ are said conditionally equivalent (denoted $u \triangleq v$) iff

$$\forall \rho \in \Sigma^* : \; p(u|\rho, ?) = p(v|\rho, ?)$$

Here, the pair $(\rho, ?)$ denotes a masked symbol surrounded by the context ρ.

Definition 3.4 (Conditionally Similar Symbols)**.** Symbols $u, v \in \Sigma$ are said conditionally similar iff

$$d([p(u|\rho, ?)]_{\rho \in \mathrm{P}}, \; [p(v|\rho, ?)]_{\rho \in \mathrm{P}}) < \epsilon$$

In other words, two conditionally equivalent/similar symbols, given any context ρ, have the same/similar probability of appearing in that context. For example, in the context

`the glass of water is half ?`

both words `empty` and `full` have the same/similar probability to appear. If this can be extended to all/most other contexts, then we would say that `empty` and `full` are conditionally equivalent/similar.

One of the most important hypotheses that shaped modern language models (Mikolov et al., 2013; Peters et al., 2018) is the distributional hypothesis (Harris, 1954) which roughly states that *similar words in meaning have similar context*. To the best of our knowledge, the distributional hypothesis is always reported informally. Here we attempt to formalize this concept:

Definition 3.5 (Strong Distributional Hypothesis)**.**

$$u \stackrel{\circ}{=} v \iff u \triangleq v$$

Definition 3.6 (Weak Distributional Hypothesis)**.**

$$d([p(y|\rho, u)]_{y \in \mathcal{Y}, \rho \in \mathrm{P}}, \; [p(y|\rho, v)]_{y \in \mathcal{Y}, \rho \in \mathrm{P}}) \leq \epsilon \iff$$
$$d([p(u|\rho, ?)]_{\rho \in \mathrm{P}}, \; [p(v|\rho, ?)]_{\rho \in \mathrm{P}}) \leq \epsilon$$

When the distributional hypothesis holds, two symbols u and v are semantically equivalent/similar if and only if u and v are conditionally equivalent/similar. In other words, one can be replaced with the other without affecting the outcome distribution iff given any context both have the same probability of being found in that context.

While we adopt the Def. 3.5 and Def. 3.6 for the distributional hypothesis, it is important to note that other interpretations exist. For example, one could weaken Def. 3.3 to account for symbols that have consistently different conditional probabilities (i.e. $u \stackrel{\circ}{=} v \iff \forall \rho \in \mathrm{P} : \; p(u|\rho) \propto p(v|\rho)$). Ultimately, our interpretation is motivated by its simplicity.

It is also important to note that, for most applications, delineating the degree to which Def. 3.5 holds true for a given dataset may prove difficult, if not infeasible. Consequently, determining whether the results discussed in this work apply to the chosen scenario may not be possible.

As we will show, when the strong distributional hypothesis (Def. 3.5) holds, then the input and output embeddings of language models, in order to be optimal, must encode the same semantic relationships, thus justifying the weight-tying technique.

4. Language Modeling

4.1. output embeddings & conditional equivalence

It is already well known that language models organize both input and output embeddings in a semantic space (Derby et al., 2020) according to intrinsic and extrinsic evaluation benchmarks (Bakarov, 2018). However, the reason why such an organization appears is still obscure.

Let us consider output embeddings. These embeddings are usually used in the following manner:

$$f(x; \theta) = \mathrm{softmax}(g(x; \theta) \cdot E^O) \tag{1}$$

Where $f(x; \theta) \in \Delta^{V-1}$ represents the Neural Network (NN) parametrized by θ.[1] Given a sequence of symbols x, $f(x; \theta)$ outputs a probability distribution over the possible symbols. $g(x; \theta) \in \mathbb{R}^d$ is a NN component that outputs an encoding vector of the input sequence. Let us use the notation $f(w|x; \theta)$ to denote the probability given to the symbol w by the model f parametrized by θ. Whenever θ is omitted, we intend the optimal model—$\forall \rho \in \mathrm{P}, w \in \Sigma : f(w|\rho) = p(w|\rho)$.

Let u and v be conditionally equivalent symbols ($u \triangleq v$). Then, for any input sequence x, we have that $p(u|x) = p(v|x)$. Thus, an optimal model will need to give the same conditional probability to u and v ($f(u|x) = f(v|x)$). To achieve this, the most natural way using Eq. 1 is to set $E^O(u) = E^O(v)$. Since f is generally not bijective there could be other values of $E^O(u)$ and $E^O(v)$ that yield $f(u|x) = f(v|x)$. However, it can be shown that there is a simple condition that yields the desired result. This is the subject of the next theorem:

Theorem 4.1 (Output Embeddings Equivalence)**.**

[1] here, Δ^k denotes the probability simplex of dimension k

1. *If there are* $x_1, \dots, x_d$ *such that matrix* $\mathcal{B} = [g(x_1), \dots, g(x_d)]$ *form a basis of* $\mathbb{R}^d$*, and*
2. *there are* $u, v \in \Sigma$ *such that* $u \triangleq v$

then $E^O(u) = E^O(v)$.

Refer to Appendix A for the proof. It is important to acknowledge that if the first hypothesis, concerning the basis $\mathcal{B}$, were absent, then $E^O(u) = E^O(v)$ would be just one of several potential solutions, resulting in $f(u|x) = f(v|x)$. Additionally, it is worth noting that the existence of the basis $\mathcal{B}$ is only possible if there exist $x_1, \dots, x_d$ such that $g(x_1), \dots, g(x_d)$ are linearly independent, equivalently, $det[g(x_1), \dots, g(x_d)] \neq 0$. However, the determinant is a polynomial of the entries $g(x_1), \dots, g(x_d)$ and has zero Lebesgue measure. Thus, a slight perturbation of $g(x_1), \dots, g(x_2)$ would establish a basis. As a consequence, the event not having a basis becomes extremely unlikely. A similar assumption is made in the context of invertible neural network, where weights matrices need to form a basis in order to be invertible (Finzi et al., 2019).

Recent works have shown that during training, embedding matrices may organize themselves into a lower-dimensional manifold (Cai et al., 2020). Importantly, this behavior does not necessarily disrupt the basis hypothesis, as the vectors in the manifold can still form a basis of $\mathbb{R}^d$, unless the manifold is entirely contained within a $d-1$-dimensional hyperplane passing through the origin.

This theorem shows that the output embeddings of an optimal language model (under Th. 4.1 hypothesis) need to have the following property: if two symbols are conditional equivalent they must have the same output embeddings. Therefore, under the strong distributional assumption (Def. 3.5) and Th. 4.1 hypothesis, the output embeddings of an optimal language model encode semantic equivalence relationships. This behavior was already empirically observed in Inan et al. (2016), Press & Wolf (2017), and Derby et al. (2020).

4.2. input embeddings & semantic equivalence

Similarly to output embeddings, during training also input embeddings organize themselves in a semantic space.

Intuitively, encoding semantically equivalent symbols with the same input embeddings seems reasonable. This approach ensures that the output of the network remains unchanged when replacing a symbol with its semantically equivalent counterpart. Recall the previous example:

`the ?  of water is half empty/full`

Since we want the network to have the same output distribution for the masked token regardless of having `empty` or `full`, then a natural way to achieve this is to encode symbols `empty` and `full` with the same vector.

While intuitive, this condition is much more difficult to show as it is architecture-dependent. Therefore, we will need to make architectural assumptions on the NN. Our assumptions are similar to those found in Tian et al. (2023). In particular, we will assume a NN made of an input embedding layer, a single self-attention layer (Vaswani et al., 2017), and an output embedding layer:

$$f(X;\theta) = \text{softmax}(g(X;\theta)E^O)$$
$$g(X;\theta) = \text{softmax}(XX^T)X$$
$$X = E^I(\rho, w;\theta)$$

where the notation $E^I(\rho, w;\theta)$ denotes the application of the input embedding layer, parametrized by θ, to every symbol in the context-symbol pair (ρ, w). The result of this application is a matrix $n \times d$, where the i-th row represents the embedding of the i-th symbol.

When such a model is optimal we can prove the following result:

Theorem 4.2 (Input Embeddings Equivalence)**.**

1. *There are symbols* $\sigma_1, \dots, \sigma_d$ *such that* $\mathcal{B} = E^O(\sigma_1, \dots, \sigma_d)$ *is a basis of* $\mathbb{R}^d$*, and*
2. *there is a symbol* s *such that the coefficients* a_i *of the linear combination* $E^O(s) = \sum_i a_i E^O(\sigma_i)$ *(such* a_i *always exists for* $\mathcal{B}$ *is a basis) do not add up to one, i.e.,* $\sum_i a_i \neq 1$*, and*
3. *there are* $u, v \in \Sigma$ *such that* $u \mathrel{\mathring{=}} v$

then

$$||E^I(u) - E^I(v)|| \leq 2 \min_\rho \left\{ \max \left\{ ||E^I(\rho, u)||, ||E^I(\rho, v)|| \right\} \right\}$$

The proof can be found in Appendix B. This result suggests that when two symbols are semantically equivalent, and under suitable conditions, an optimal model must encode these symbols with embeddings that are close to each other. Similarly to Th. 4.1 the basis hypothesis should not pose significant challenge. Finally, while we consider a simplified case, works such of Haider et al. (2023) (characterizing the injectivity profile of the ReLU activation) and of Morris et al. (2023) (successfully inverting a foundational language model with high accuracy) suggest that this theorem may hold for more general architectures.

4.3. Weight Tying

Both Theorems 4.1 and 4.2 demonstrate that, under the respective conditions, an optimal model must encode conditional equivalence relationships in the output embedding matrix and semantic equivalence relationships in the input

embedding matrix. However, assuming the distributional hypothesis (Def. 3.5), we observe that conditional equivalence encoded in the output matrix is equivalent to semantic equivalence encoded in the input matrix. Consequently, tying the weights of input and output embeddings results in encoding the same information through both semantic equivalence and conditional equivalence of the underlying data.

In summary, when the distributional hypothesis holds, both input and output embeddings need to exhibit similar relationships. Therefore, tying their weights should enhance training. However, when the distributional hypothesis does not hold, input and output embeddings may attempt to encode different relationships. In such cases, tying their weights could have a disruptive impact, as the relationships between output and input embeddings need to differ.

To see this, Let u and v be semantically equivalent symbols. Therefore, by Theorem 4.2, we know that $E^I(u)$ will be close to $E^I(v)$. Now, suppose also that u and v are not to be conditionally equivalent, but rather conditionally different. For example, we have that $\forall \rho : |p(u|\rho, ?) - p(y|\rho, ?)| > 1 - \epsilon$. This means that an optimal model will have $E^O(u)$ far apart from $E^O(v)$. As you can see, if we are to tie the embedding layers ($E^I = E^O$) we could run easily into a problem as the same embeddings need to be close and far apart at the same time.

Finally, we note that weight tying may offer different properties than those highlighted in this work. Consequently, the technique may prove helpful or detrimental even under the conditions discussed here.

To summarize Theorems 4.1 and 4.2 suggest the following behaviors:

- semantically equivalent symbols should have input embeddings close to each other,
- conditionally equivalent symbols should have output embedding close to each other,
- and input and output embedding weight tying is beneficial under distributional hypothesis and detrimental otherwise.

5. Experiments

Overview. As previously mentioned, the fact that foundational language models organize both their input and output embeddings in a semantically relevant space has already been observed (Derby et al., 2020). Instead, we aim to test the behaviors predicted by Theorems 4.1, and 4.2 in a small and controlled scenario where these behaviors can be clearly observed.

EXor Problem. To test the NN behaviors, we will use an extended version of the Xor problem—denoted EXor. In the traditional Xor problem, we are given a binary string, and the model is required to predict whether the sum of its digits is even or not. For example, the string `010011` is odd, while the string `011101` is even. In our extension, we formalize this problem as a language modeling problem. The model is given strings in the form of `0011?01E` and it is asked to fill the masked token. Here, symbol `E` means that the binary string is even, therefore we know that the masked symbol was a `1` symbol. On the other hand, the symbol `D` denotes an odd binary string. Therefore given an input-output example `011?01D`, we know that the masked token is `0`. Note that the last symbol cannot be subject to masking so examples such as `110011?` cannot happen.

Semantically Equivalent Symbols in EXor. Further, to simulate semantically equivalent symbols, we use two different symbols to represent `0`—$\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$; and two different symbols to represent `1`—$\texttt{1}_\texttt{A}$ and $\texttt{1}_\texttt{B}$. Therefore, samples like $\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{0}_\texttt{A}\texttt{0}_\texttt{A}\texttt{1}_\texttt{B}\texttt{?E}$ should always result in either $\texttt{1}_\texttt{B}$ or $\texttt{1}_\texttt{A}$. Meanwhile, samples like $\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{?}\texttt{0}_\texttt{A}\texttt{1}_\texttt{B}\texttt{1}_\texttt{B}\texttt{E}$ may result in both $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$.

Conditionally Equivalent Symbols in EXor. To simulate conditionally equivalent symbols, we use the same probability for the symbols $\texttt{1}_\texttt{A}$ and $\texttt{1}_\texttt{B}$. Therefore, the probability $p(\texttt{?} = \texttt{1}_\texttt{A}|\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{0}_\texttt{A}\texttt{0}_\texttt{A}\texttt{?}\texttt{1}_\texttt{B}\texttt{E}) = p(\texttt{?} = \texttt{1}_\texttt{B}|\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{0}_\texttt{A}\texttt{0}_\texttt{A}\texttt{?}\texttt{1}_\texttt{B}\texttt{E}) = 1/2$. Since $\texttt{1}_\texttt{A}$ and $\texttt{1}_\texttt{B}$ are both conditionally ($\texttt{1}_\texttt{A} \triangleq \texttt{1}_\texttt{B}$) and semantically equivalent ($\texttt{1}_\texttt{A} \triangleq \texttt{1}_\texttt{B}$), we have that the distributional hypothesis holds for this specific pair. Therefore, during training their input and output embeddings should become close to each other.

Conditionally Different Symbols in EXor. Instead, to simulate conditionally different symbols, we can use different probabilities for the symbols $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$. Therefore, the probability $p(\texttt{?=}\texttt{0}_\texttt{A}|\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{0}_\texttt{A}\texttt{?}\texttt{1}_\texttt{B}\texttt{1}_\texttt{B}E) = \epsilon << 1 - \epsilon = p(\texttt{?} = \texttt{0}_\texttt{B}|\texttt{1}_\texttt{A}\texttt{1}_\texttt{B}\texttt{0}_\texttt{A}\texttt{?}\texttt{1}_\texttt{B}\texttt{1}_\texttt{B}\texttt{E})$. Since $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$ are not conditionally equivalent ($\texttt{0}_\texttt{A} \not\triangleq \texttt{0}_\texttt{B}$) while being semantically equivalent ($\texttt{0}_\texttt{A} \triangleq \texttt{0}_\texttt{B}$), we have that the distributional hypothesis does not hold. Therefore, during training, only their input embeddings should become close to each other.

Instead, note that pairs such as $\texttt{0}_\texttt{A}$ and $\texttt{1}_\texttt{B}$ are neither semantic equivalent ($\texttt{0}_\texttt{A} \not\triangleq \texttt{1}_\texttt{B}$) nor conditionally equivalent ($\texttt{0}_\texttt{A} \not\triangleq \texttt{1}_\texttt{B}$). A few examples of input and outputs are provided in Table 1

Dataset. The training split of the EXor problem is generated by considering 90% of all possible binary strings of size N. Then, even binary strings are concatenated with the symbol `E`, while odd strings are concatenated with the symbol `D`. Further, `1` symbols are replaced with either $\texttt{1}_\texttt{A}$ or $\texttt{1}_\texttt{B}$ with probability $1/2$ respectively. `0` symbols are replaced with $\texttt{0}_\texttt{A}$ with probability $1/10$ or with $\texttt{0}_\texttt{B}$ with probability $9/10$. The test set is generated similarly using the remaining 10% of the binary string of size N. In our experiments, we

x	$p(?\|x)$	
$0_B 0_A 0_A 0_B ? 0_A E$	$p(0_A\|x) = \epsilon,$	$p(0_B\|x) = 1-\epsilon$
$1_A ? 0_A 0_A 0_B 1_A D$	$p(1_A\|x) = \frac{1}{2},$	$p(1_B\|x) = \frac{1}{2}$
$1_A 1_B ? 0_A 0_B 1_A D$	$p(0_A\|x) = \epsilon,$	$p(0_B\|x) = 1-\epsilon$
$0_B 0_A 0_A 0_B ? 0_A D$	$p(1_A\|x) = \frac{1}{2},$	$p(1_B\|x) = \frac{1}{2}$

Table 1. Input examples and resulting output distribution.

set $N = 7$.

Architecture. The model architecture is similar to the one assumed in Theorem 4.2. We used a single layer, single head, with gelu (Hendrycks & Gimpel, 2016) activation, Transformer Encoder (Vaswani et al., 2017) architecture from the PyTorch API[2]. The input embedding layers has vocabulary size 7 (1_A, 1_B, 0_A, 0_B, E, D, ?) where each embedding size is 4 (i.e., $E^I \in \mathbb{R}^{7\times4}$, $E^O \in \mathbb{R}^{4\times7}$). Excluded the masked token, among the 7 symbols only 4 have a semantically distinct meaning ($1_A \triangleq 1_B$, $0_A \triangleq 0_B$, E, D). Since, from hypotheses of Theorems 4.1 and 4.2, we want the input and output embedding matrix to be a basis for $\mathbb{R}^d$, it is necessary to keep $d \leq 4$. We choose $d = 4$. However, in Appendix C we experiment with larger architectures. Finally, the model with tied/untied input and output embeddings is referred to as tied/untied model.

Hyperpameters. We used AdamW (Loshchilov & Hutter, 2018) (PyTorch implementation[3]) optimizer with 5e−4 learning rate, 1e−1 weight decay. Further, we employ a cosine learning rate scheduler (PyTorch implementation[4]) with 1e−5 minimum learning rate and 1e4 iteration cycle. The batch size is 114 (size of the training split). We train for 1.5e5 iterations.

5.1. Results

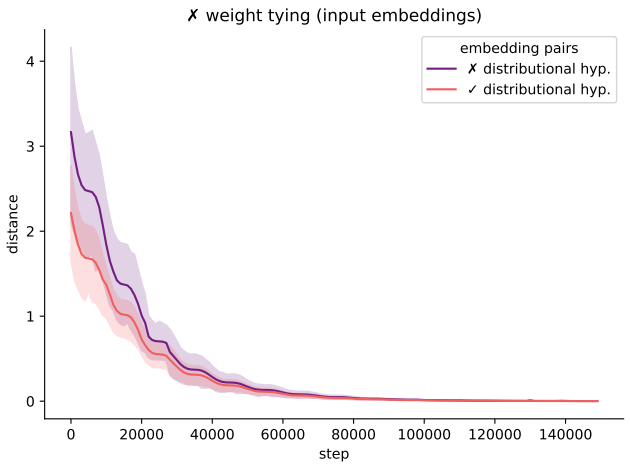
Results are displayed in Fig. 1 and 2. We show the mean and the 95% confidence interval of 5 repetitions of the same experiment.

Untied Input Embeddings. Let us consider Fig. 1a which displays the input embedding distances between 1_A vs. 1_B (✓ distributional hyp.) and 0_A vs. 0_B (✗ distributional hyp.) for the untied model. Note that, 1_A is semantically equivalent to 1_B and 0_A is semantically equivalent to 0_B. Therefore, from Theorem 4.2, we expect the input embeddings of 1_A and 1_B to become close to each other. The same goes for 0_A and 0_B. Fig. 1a confirms the tendency to en-

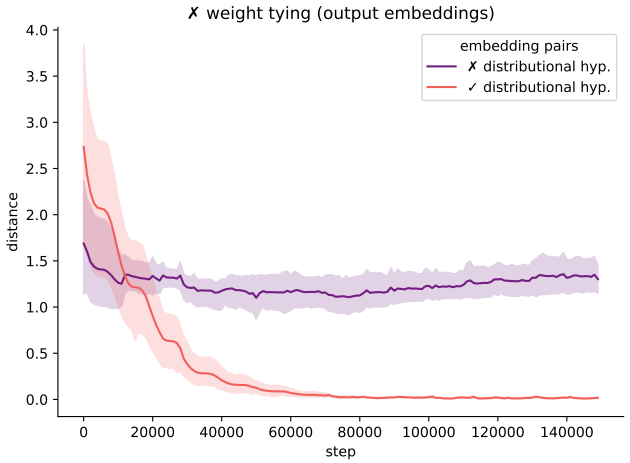
[2] https://pytorch.org/docs/2.1/generated/torch.nn.TransformerEncoderLayer.html

[3] https://pytorch.org/docs/2.1/generated/torch.optim.AdamW.html

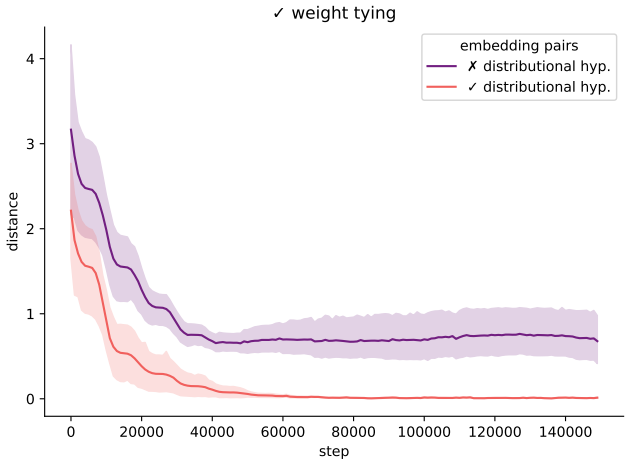
[4] https://pytorch.org/docs/2.1/generated/torch.optim.lr_scheduler.CosineAnnealingLR.html



(a) Input embeddings distances for the EXor problem with untied weights



(b) Output embeddings distances for the EXor problem with untied weights



(c) Embeddings distances for the EXor problem with tied weights

Figure 1. Embedding distances between symbols when the distributional hypothesis holds and does not hold. The distance used is the Euclidean. The line plot is the result of the average of 5 runs. With lower opacity, we display the 95% confidence interval

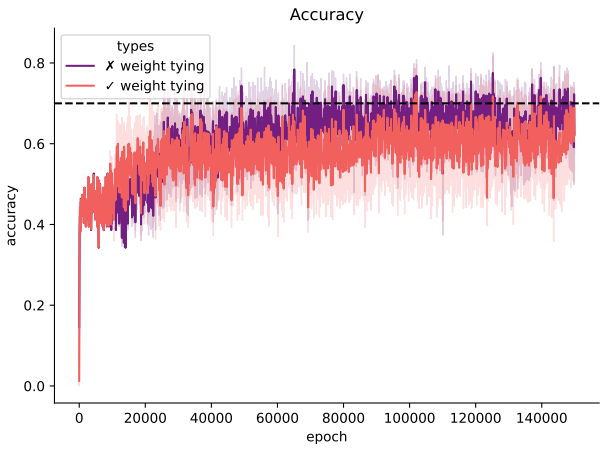


Figure 2. Test accuracies for the EXor problem with and without tied embedding weights. The dashed line denotes the optimal accuracy achievable. The line plot displays the mean and 95% confidence interval of 5 runs

code semantically equivalent symbols with close embedding representations.

Untied Output Embeddings. Let us consider Fig. 1b which displays the output embedding distances of $\mathtt{1}_\mathtt{A}$ vs. $\mathtt{1}_\mathtt{B}$ (✓ distributional hyp.) and $\mathtt{0}_\mathtt{A}$ vs. $\mathtt{0}_\mathtt{B}$ (✗ distributional hyp.) for the untied model. Here, only $\mathtt{1}_\mathtt{A}$ and $\mathtt{1}_\mathtt{B}$ are conditionally equivalent. From Theorem 4.1, we expect only output embeddings of $\mathtt{1}_\mathtt{A}$ and $\mathtt{1}_\mathtt{B}$ to be close to each other. Fig. 1b confirms this trend as only the pair $\mathtt{1}_\mathtt{A}$-$\mathtt{1}_\mathtt{B}$ (✓ distributional hyp.) approaches distance 0.

Tied Embeddings. Instead, Fig. 1c displays the input/output embedding distances of $\mathtt{1}_\mathtt{A}$ vs. $\mathtt{1}_\mathtt{B}$ (✓ distributional hyp.) and $\mathtt{0}_\mathtt{A}$ vs. $\mathtt{0}_\mathtt{B}$ (✗ distributional hyp.) for the tied model. Since $\mathtt{1}_\mathtt{A}$ and $\mathtt{1}_\mathtt{B}$ are both semantically and conditionally equivalent, then the distributional hypothesis 3.5 hold. In this case, by combining Theorems 4.1 and 4.2, we expect that $\mathtt{1}_\mathtt{A}$ and $\mathtt{1}_\mathtt{B}$ embeddings will converge to the same vector faster compared to the untied model. This can be verified by considering Fig. 1c in which $\mathtt{1}_\mathtt{A}$ and $\mathtt{1}_\mathtt{B}$ embeddings become extremely close faster compared to Fig. 1a. Instead, since $\mathtt{0}_\mathtt{A}$ and $\mathtt{0}_\mathtt{B}$ are not conditionally equivalent, then the distributional hypothesis does not hold. Therefore, by combining Theorems 4.1, and 4.2 and Eq. 3.5, we can expect their embeddings distance to be subject to high variance, as from one side they need to be close to each other, and from the other side they need to be far from each other. Fig. 1c confirms this behavior.

Accuracy. Fig. 2 displays the test accuracy for the untied model (✗ weight tying) and the tied one (✓ weight tying). Since for the current setting of the EXor problem, the distributional hypothesis does not hold for all the symbols, we would expect a certain degree of instability in the training of the tied model. This is because, embeddings of $\mathtt{0}_\mathtt{A}$ and $\mathtt{0}_\mathtt{B}$ are pulled close to each other (since $\mathtt{0}_\mathtt{A} \stackrel{\circ}{=} \mathtt{0}_\mathtt{B}$), and pushed away from each other (since $\mathtt{0}_\mathtt{A} \stackrel{\Delta}{\neq} \mathtt{0}_\mathtt{B}$). This instability is reflected in the slightly lower-performing model wrt. the untied model. It can also be noted that the untied model achieves a higher level of accuracy approaching the optimal value denoted by the dashed black line.

5.2. Discussion

We are finally ready to answer the RQs proposed in Sect. 1.

RQ_1 Why the input embeddings in foundation models encode semantic information?

Our findings suggest that the semantic information encoded by input embeddings does align with the notion of semantically equivalent symbols. Intuitively, if two symbols are semantically equivalent (Def. 3.1) then they will be close in input embedding space. This was the subject of Theorem 4.2 and it was empirically verified in Fig. 1a.

RQ_2 Why the output embeddings in foundation models encode semantic information?

Our findings suggest that the semantic information encoded by output embeddings does align with the notion of conditionally equivalent symbols. Intuitively, if two symbols are conditionally equivalent (Def. 3.3) then they will be close in output embedding space. This was the subject of Theorem 4.1 and it was empirically verified in Fig. 1b.

RQ_3 Why input-output embeddings weight tying is effective?

Our findings suggest that under the distributional hypothesis (Def. 3.5) both input and output embeddings encode the same relationships. In the input embeddings, these relationships are encoded by exploiting semantic equivalences in the data. In the output embeddings, these relationships are encoded by exploiting conditional equivalences in the data. However, since conditional equivalences coincide with semantical equivalences, it appears reasonable to tie the weights of the embeddings. This is consideration resulting from the combination of Theorems 4.1, 4.2, and Definition 3.5.

We also note that it has been observed that foundation models that overfit early in the validation set and are continuously trained for several more epochs achieve better human ratings (Ouyang et al., 2022). While the mechanisms behind this behavior are still unknown, we can observe that also in our experiments perfect alignment between semantically equivalent and distributional equivalent symbols happens only several epochs after validation overfitting.

Not all agree on the fact that weight tying is always beneficial in NLP language models (Chung et al., 2020). We

believe that this may be caused by the fact that the Distributional Hypothesis does not hold strictly like in Def. 3.5, but more like in its weaker form Def. 3.6 for the NLP domain. Therefore, it may be beneficial only early in training when relationships between symbols are established grossly, and detrimental later when these relationships are more subtle.

Of course, NN training is a very chaotic process that rarely ends up in an optimum. Therefore, these results may not hold empirically in all scenarios (Appendix C is devoted to validating the theorems on different architectures and datasets). However, we believe that practitioners, in light of these findings, should carefully decide when using tied input and output embeddings, as if the distributional hypothesis does not hold it may result in training instabilities. On the other hand, when the distributional hypothesis does hold, then the training should become faster. The main takeaway of this work for practitioners can be summarized in:

Use tied input-output embeddings only when the distributional hypothesis hold

6. Related Works

The literature regarding the analysis of word embeddings is extensive. Here, we only aim to provide a brief overview of foundational and recent advances in this scientific direction.

Embedding Generation. While we discuss exclusively non-contextual embeddings arising from foundational language models a substantial body of research discusses techniques for generating semantically relevant non-contextual embeddings (Mikolov et al., 2013; Pennington et al., 2014; Vilnis & McCallum, 2014; Bojanowski et al., 2017) and contextual embeddings (Melamud et al., 2016; McCann et al., 2017; Peters et al., 2018; Radford et al., 2019; Wang et al., 2021), refer to (Almeida & Xexèo, 2019) and (Torregrossa et al., 2021) for a review. Recently, Qin & Van Durme (2023) proposed a method to generate embeddings from sentence snippets, and Bai et al. (2023) proposed a method using optimal transport theory.

Embedding Evaluation. While our findings suggests that the quality of embeddings should be measured in terms of semantical and conditional equivalence, an extensive literature is dedicated to evaluating their intrinsic and extrinsic properties. Intrinsic properties compare the embeddings wrt. human judgment such as direct word comparison (Baroni et al., 2014; Hill et al., 2015; Gerz et al., 2016), word analogies (Pereira et al., 2016; Gladkova et al., 2016), and thematic fit (Sayeed et al., 2016). Extrinsic properties measure the embedding performance on downstream tasks such as text classification (Tsvetkov et al., 2015), Sentiment Analysis (Schnabel et al., 2015; Zhou et al., 2016), Name Entity Recognition (Turian et al., 2010), and Semantic Role Labeling (Ettinger et al., 2016). Refer to (Bakarov, 2018) and (Torregrossa et al., 2021) for a review.

Embedding Analysis. Some works explore the theoretical properties of embedding generation algorithms. A foundation work (Levy & Goldberg, 2014) connects skip-gram with negative-sampling to the pointwise mutual information. A geometric analysis of the same word modeling technique is provided in (Mimno & Thompson, 2017). Yin & Shen (2018) studied the dimensionality of word embeddings.

Embedding Information. While we do not focus on the inner properties encoded in the embeddings many explored the inner information embedded in the vector representations of language models (Köhn, 2015; Adi et al., 2017; Hupkes et al., 2018). For comprehensive overviews, refer to (Rogers et al., 2021; Belinkov, 2022). Much of this literature concentrates on investigating whether language models encode syntactic information such as part-of-speech tagging (Shi et al., 2016; Belinkov et al., 2017), syntactic number (Giulianelli et al., 2018), or sentence structure (Liu et al., 2019; Hewitt & Manning, 2019; Lin et al., 2019). Several studies have delved into decoding semantic properties, including entity attributes (Gupta et al., 2015; Grand et al., 2022), sentiment analysis (Radford et al., 2017), semantic role labeling (Ettinger et al., 2016; Tenney et al., 2018), world state representation (Li et al., 2021), and agent property identification (Andreas, 2022). Other works have addressed the semantic property of topic structure (Meng et al., 2022; Zhang et al., 2022; Li et al., 2023b).

7. Conclusion & Future Works

In this work, we analyzed the effect of weight tying on input and output embeddings both from a theoretical and empirical perspective. Firstly, to study the theoretical implications, we formalized the distributional hypothesis using the notions of semantical and conditional equivalence. Next, we demonstrated that semantically equivalent symbols are encoded in similar input embeddings, and conditionally equivalent symbols are encoded in similar output embeddings. Thus, we concluded that one should consider the practice of tying embeddings only when the distributional hypothesis holds, at least in its weak form. Finally, we supported the theoretical findings with a battery of experiments on the EXor problem. Additionally, we note that exploring alternative interpretations of the distributional hypothesis may lead to further insight into the semantic space of embeddings. The code for reproducing the experiments is available at:

https://zenodo.org/records/11103163

Acknowledgment

This work was funded by the MIUR projects "T-LADIES" (PRIN 2020TL3X8X). We thank Federica Bertolotti And

Luca Favalli for proof reading Theorems 4.1 and 4.2.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

References

Adi, Y., Kermany, E., Belinkov, Y., Lavi, O., and Yoav, G. Fine-Grained Analysis of Sentence Embeddings Using Auxiliary Prediction Tasks. In Larochelle, H., Vinyals, O., and Sainath, T. (eds.), *Proceedings of the 5th International Conference on Learning Representations (ICLR'17)*, Toulon, France, April 2017.

Almeida, F. and Xexèo, G. Word Embeddings: A Survey. *arXiv e-prints*, arXiv:1901.09069:1–11, January 2019.

Andreas, J. Language Models as Agent Models. In Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP'22)*, pp. 5798–5808, Abu Dhabi, December 2022. Association for Computational Linguistics.

Bai, Y., Medri, I. V., Martin, R. D., Shahroz, R., and Kolouri, S. Linear Optimal Partial Transport Embedding. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, pp. 1492–1520, Honolulu, Hawaii, July 2023. PMLR.

Bakarov, A. A Survey of Word Embeddings Evaluation Methods. *arXiv e-prints*, arXiv:1801.09536:1–26, January 2018.

Baroni, M., Dinu, G., and Kruszewski, G. Don't Count, Predict! A Systematic Comparison of Context-Counting vs. Context-Predicting Semantic Vectors. In Toutanova, K. and Wu, H. (eds.), *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL'14)*, pp. 238–247, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

Belinkov, Y. Probing Classifiers: Promises, Shortcomings, and Advances. In Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP'22)*, pp. 207–219, Abu Dhabi, December 2022. Association for Computational Linguistics.

Belinkov, Y., Durrani, N., Dalvi, F., Sajjad, H., and Glass, J. What Do Neural Machine Translation Models Learn about Morphology? In Barzilay, R. and Kan, M.-Y. (eds.), *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL'17)*, pp. 861–872, Vancouver, Canada, July 2017. Association for Computational Linguistics.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, June 2017.

Cai, X., Huang, J., Bian, Y., and Church, K. Isotropy in the Contextual Embedding Space: Clusters and Manifolds. In Song, D., Cho, K., and White, M. (eds.), *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, Addis Ababa, Ethiopia, April 2020.

Chen, J., Ma, M., Zheng, R., and Huang, L. MAM: Masked Acoustic Modeling for End-to-End Speech-to-Text Translation. *arXiv e-prints*, arXiv:2010.11445:1–12, October 2020.

Cho, J., Lei, J., Tan, H., and Bansal, M. Unifying Vision-and-Language Tasks via Text Generation. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning (ICML'21)*, pp. 1931–1942. PMLR, July 2021.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. PaLM: Scaling Language Modeling with Pathways. *Journal of Machine Learning Research*, 24(240):1–113, August 2023.

Chung, H. W., Fevry, T., Tsai, H., Johnson, M., and Ruder, S. Rethinking Embedding Coupling in Pre-Trained Language Models. In Song, D., Cho, K., and White, M. (eds.), *Proceedings of the 8th International Conference on Learning Representations (ICLR'20)*, Addis Ababa, Ethiopia, April 2020.

Derby, S., Miller, P., and Devereux, B. Analysing Word Representation from the Input and Output Embeddings in Neural Network Language Models. In Fernández, R. and Linzen, T. (eds.), *Proceedings of the 24th Conference on Computational Natural Language Learning (CoNLL'20)*,

pp. 442–454. Association for Computational Linguistics, November 2020.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 17th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT'19)*, pp. 4171–4186, Minneapolis, MN, USA, June 2019. Association for Computational Linguistics.

Ettinger, A., Elgohary, A., and Resnik, P. Probing for Semantic Evidence of Composition by Means of Simple Classification Tasks. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP (RepEval'16)*, pp. 134–139, Berlin, Germany, August 2016. Association for Computational Linguistics.

Finzi, M., Izmailov, P., Maddox, W., Kirichenko, P., and Wilson, A. G. Invertible Convolutional Networks. In *Proceedings of the 1st Workshop on Invertible Neural Nets and Normalizing Flows (INNNF'19)*, Long Beach, CA, USA, June 2019. PMLR.

Firth, J. R. A Synopsis of Linguistic Theory, 1930–1955. In Firth, J. R. (ed.), *Studies in Linguistic Analysis*, pp. 1–31. Blackwell, Oxford, United Kingdom, 1957.

Gerz, D., Vulić, I., Hill, F., Reichart, R., and Korhonen, A. SimVerb-3500: A Large-Scale Evaluation Set of Verb Similarity. In Su, J., Duh, K., and Carreras, X. (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*, pp. 2173–2182, Austin, TX, USA, September 2016. Association for Computational Linguistics.

Giulianelli, M., Harding, J., Mohnert, F., Hupkes, D., and Zuidema, W. Under the Hood: Using Diagnostic Classifiers to Investigate and Improve How Language Models Track Agreement Information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP'18)*, pp. 240–248, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

Gladkova, A., Drozd, A., and Matsuoka, S. Analogy-Based Detection of Morphological and Semantic Relations with Word Embeddings: What Works and What Doesn't. In Andreas, J., Choi, E., and Lazaridou, A. (eds.), *Proceedings of the NAACL Student Research Workshop (NAACL'16)*, pp. 8–15, San Diego, CA, USA, June 2016. Association for Computational Linguistics.

Grand, G., Blank, I. A., Pereira, F., and Fedorenko, E. Semantic Projection Recovers Rich Human Knowledge of Multiple Object Features from Word Embeddings. *Nature Human Behavior*, 6:975–987, April 2022.

Gunasekar, S., Zhang, Y., Aneja, J., Cesar, C., Mendes, T., Del Giorno, A., Gopi, S., Javaheripi, M., Kauffmann, P., de Rosa, G., Saarikivi, O., Salim, A., Shah, S., Singh Behl, H., Wang, X., Bubeck, S., Eldan, R., Kalai, A. T., Lee, Y. T., and Li, Y. Textbooks Are All You Need. *arXiv e-prints*, arXiv:2306.11644:1–26, June 2023.

Gupta, A., Boleda, G., Baroni, M., and Padó, S. Distributional Vectors Encode Referential Attributes. In Màrquez, L., Callison-Burch, C., and Su, J. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, pp. 12–21, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Haider, D., Ehler, M., and Balazs, P. Convex Geometry of ReLU-Layers, Injectivity on the Ball and Local Reconstruction. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, pp. 12339–12350, Honolulu, HA, USA, July 2023. PMLR.

Harris, Z. S. Distributional Structure. *WORD*, 10(2–3): 146–162, 1954.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked Autoencoders Are Scalable Vision Learners. In Dana, K., Hua, G., Roth, S., Samaras, D., and Singh, R. (eds.), *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'22)*, pp. 15979–15988, New Orleans, LA, USA, June 2022. IEEE.

Hendrycks, D. and Gimpel, K. Gaussian Error Linear Units (GELUs). *arXiv e-prints*, arXiv:1606.08415:1–9, June 2016.

Hewitt, J. and Manning, C. D. A Structural Probe for Finding Syntax in Word Representations. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'19)*, pp. 4129–4138, Minneapolis, MN, USA, June 2019. Association for Computational Linguistics.

Hill, F., Reichart, R., and Korhonen, A. Simlex-999: Evaluating Semantic Models with (Genuine) Similarity Estimation. *Computational Linguistics*, 41(4):665–695, December 2015.

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

Hupkes, D., Veldhoen, S., and Zuidema, W. Visualisation and 'Diagnostic Classifiers' Reveal How Recurrent and Recursive Neural Networks Process Hierarchical Structure. *Journal of Artificial Intelligence Research*, 61:907–926, 2018.

Inan, H., Khosravi, K., and Socher, R. Tying Word Vectors and Word Classifiers: A Loss Framework for Language Modeling. In Bengio, S. and Kingsbury, B. (eds.), *Proceedings of the 4th International Conference on Learning Representations (ICLR'16)*, San Juan, Puerto Rico, May 2016.

Köhn, A. What's in an Embedding? Analyzing Word Embeddings through Multilingual Evaluation. In Màrquez, L., Callison-Burch, C., and Su, J. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, pp. 2067–2073, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In Rush, A. (ed.), *Proceedings of the 7th International Conference on Learning Representations (ICLR'19)*, New Orleans, LA, USA, May 2019.

Levine, Y., Lenz, B., Dagan, O., Ram, O., Padnos, D., Sharir, O., Shalev-Shwartz, S., Shashua, A., and Shoham, Y. SenseBERT: Driving Some Sense into BERT. In Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL'20)*, pp. 4656–4667, Seattle, WA, USA, July 2020. Association for Computational Linguistics.

Levy, O. and Goldberg, Y. Neural Word Embedding as Implicit Matrix Factorization. In Ghahramani, Z., Welling, M., Cortes, C.and Lawrence, N., and Weinberger, K. Q. (eds.), *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14)*, pp. 2177–2185, Montréal, Canada, December 2014. Curran Associates Inc.

Li, B. Z., Nye, M., and Andreas, J. Implicit Representations of Meaning in Neural Language Models. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL'21+NLP'21)*, pp. 1813–1827. Association for Computational Linguistics, August 2021.

Li, S., Zhang, L., Wang, Z., Wu, D., Wu, L., Liu, Z., Xia, J., Tan, C., Liu, Y., Sun, B., and Li, S. Z. Masked Modeling for Self-supervised Representation Learning on Vision and Beyond. *arXiv e-prints*, arXiv:2401.00897:1–27, December 2023a.

Li, Y., Xu, L., Tian, F., Jiang, L., Zhong, X., and Chen, E. Word Embedding Revisited: A New Representation Learning and Explicit Matrix Factorization Perspective. In Yang, Q. and Wooldridge, M. (eds.), *Proceedings of the 24th International Conference on Artificial Intelligence (IJCAI'15)*, pp. 3650–3656, Buenos Aires, Argentina, July 2015. ACM.

Li, Y., Li, Y., and Risteski, A. How Do Transformers Learn Topic Structure: Towards a Mechanistic Understanding. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, pp. 19689–19729, Honolulu, Hawaii, July 2023b. PMLR.

Lin, Y., Tan, Y. C., and Frank, R. Open Sesame: Getting inside BERT's Linguistic Knowledge. In Linzen, T., Chrupała, G., Belinkov, Y., and Hupkes, D. (eds.), *Proceedings of the 2019 ACL Workshop on Analyzing and Interpreting Neural Networks for NLP (BlackboxNLP'19)*, pp. 241–253, Firenze, Italy, August 2019. Association for Computational Linguistics.

Liu, N. F., Gardner, M., Belinkov, Y., Peters, M. E., and Smith, N. A. Linguistic Knowledge and Transferability of Contextual Representations. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'19)*, pp. 1073–1094, Minneapolis, MN, USA, June 2019. Association for Computational Linguistics.

Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. In Ranzato, M. (ed.), *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*, Vancouver, BC, Canada, April/May 2018.

McCann, B., Bradbury, J., Xiong, C., and Socher, R. Learned in Translation: Contextualized Word Vectors. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, pp. 6297–6308, Long Beach, CA, USA, December 2017. Curran Associates, Inc.

Melamud, O., Goldberger, J., and Dagan, I. context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In Riezler, S. and Goldberg, Y. (eds.), *Proceedings of the 20th Conference on Computational Natural Language Learning (CoNLL'16)*, pp. 51–61, Berlin, Germany, August 2016. Association for Computational Linguistics.

Meng, Y., Zhang, Y., Huang, J., Zhang, Y., and Han, J. Topic Discovery via Latent Space Clustering of Pretrained Language Model Representations. In *Proceedings of the ACM Web Conference (WWW'22)*, pp. 3143–3152, Lyon, France, April 2022. ACM.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. Distributed Representations of Words and Phrases and Their Compositionality. In Burges, C. J. C., Bottou, L., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Proceedings of the 26th International Conference on Neural Information Processing Systems (NIPS'13)*, pp. 3111–3119, Lake Tahoe, NV, USA, December 2013. Curran Associates Inc.

Mimno, D. and Thompson, L. The Strange Geometry of Skip-Gram with Negative Sampling. In Palmer, M., Hwa, R., and Riedel, S. (eds.), *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP'17)*, pp. 2873–2878, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

Morris, J. X., Zhao, W., Chiu, J. T., Shmatikov, V., and Rush, A. M. Language Model Inversion. *arXiv e-prints*, arXiv:2311.13647:1–21, November 2023.

Nguyen, T. Q. and Salazar, J. Transformers without Tears: Improving the Normalization of Self-Attention. In Niehues, J., Cattoni, R., Stüker, S., Negri, M., Turchi, M., Ha, T.-L., Salesky, E., Sanabria, R., Barrault, L., Specia, L., and Federico, M. (eds.), *Proceedings of the 16th International Conference on Spoken Language Translation (IWSLT'19)*. Association for Computational Linguistics, November 2019.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training Language Models to Follow Instructions with Human Feedback. In Argawal, A., Oh, A., Belgrave, D., and Cho, K. (eds.), *Processing of the 35th Conference on Neural Information Processing Systems (NeurIPS'22)*, pp. 27730–27744, New Orleans, USA, November 2022. Curran Associates, Inc.

Pappas, N., Miculicich, L., and Henderson, J. Beyond Weight Tying: Learning Joint Input-Output Embeddings for Neural Machine Translation. In *Proceedings of the 3rd Conference on Machine Translation (WMT'18)*, pp. 73–83, Brussels, Belgium, October 2018. Association for Computational Linguistics.

Pennington, J., Socher, R., and Manning, C. GloVe: Global Vectors for Word Representation. In Pang, B. and Daelemans, W. (eds.), *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP'14)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

Pereira, F., Gershman, S., Ritter, S., and Botvinick, M. A Comparative Evaluation of Off-the-Shelf Distributed Semantic Representations for Modelling Behavioural Data. *Cognitive Neuropsychology*, 33(3–4):175–190, August 2016.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. Deep Contextualized Word Representations. In Walker, M., Ji, H., and Stent, A. (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'18)*, pp. 2227–2237, New Orleans, Louisiana, USA, June 2018. Association for Computational Linguistics.

Press, O. and Wolf, L. Using the Output Embedding to Improve Language Models. In *Proceedings of the 15th Conference of the European Chapter of the Association fo Computational Linguistics (EuACL'17)*, pp. 157–163, Valencia, Spain, April 2017. Association for Computational Linguistics.

Qin, G. and Van Durme, B. Nugget: Neural Agglomerative Embeddings of Text. In *Proceedings of the 40th International Conference on Machine Learning (ICML'23)*, pp. 28337–28350, Honolulu, HA, USA, July 2023. PMLR.

Radford, A., Jozefowicz, R., and Sutskever, I. Learning to Generate Reviews and Discovering Sentiment. *arXiv e-prints*, arXiv:1704.01444:01–09, April 2017.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language Models Are Unsupervised Multitask Learners. OpenAI blog, 2019.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zou, Y., Li, W., and Liu, P. J. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21 (1):5485–5551, January 2020.

Rogers, A., Kovaleva, O., and Rumshisky, A. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2021.

Rosin, G. D., Guy, I., and Radinsky, K. Time Masking for Temporal Language Models. In Akoglu, L., Dong, X. L., and Tang, J. (eds.), *Proceedings of the 15th ACM International Conference on Web Search and Data Mining (WSDM'22)*, pp. 833–841. ACM, February 2022.

Sayeed, A., Greenberg, C., and Demberg, V. Thematic Fit Evaluation: An Aspect of Selectional Preferences. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP (RepEval'16)*, pp. 99–105, Berlin, Germany, August 2016. Association for Computational Linguistics.

Schnabel, T., Labutov, I., Mimno, D., and Joachims, T. Evaluation Methods for Unsupervised Word Embeddings. In Màrquez, L., Callison-Burch, C., and Su, J. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, pp. 298–307, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Scott, D. and Strachey, C. *Toward a Mathematical Semantics for Computer Languages*. Oxford University, Oxford, United Kingdom, August 1971.

Shi, X., Padhi, I., and Knight, K. Does String-Based Neural MT Learn Source Syntax? In Su, J., Duh, K., and Carreras, X. (eds.), *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP'16)*, pp. 1526–1534, Austin, TX, USA, September 2016. Association for Computational Linguistics.

Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R. T., Kim, N., Van Durme, B., Bowman, S. R., Das, D., and Pavlik, E. What Do You Learn from Context? Probing for Sentence Structure in Contextualized Word Representations. In Ranzato, M. (ed.), *Proceedings of the 6th International Conference on Learning Representations (ICLR'18)*, Vancouver, BC, Canada, April/May 2018.

Tian, Y., Wang, Y., Chen, B., and Du, S. Scan and Snap: Understanding Training Dynamics and Token Composition in 1-Layer Transformer. *arXiv e-prints*, arXiv:2305.16380:1–37, October 2023.

Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., Lucic, M., and Dosovitskiy. MLP-Mixer: An all-MLP Architecture for Vision. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Lian, P. S., and Wortman Vaughan, J. (eds.), *Processing of the 34th Conference on Neural Information Processing Systems (NeurIPS'21)*, volume 34, pp. 24261–24272, Virtual, December 2021. Curran Associates, Inc.

Torregrossa, F., Allesiardo, R., Claveau, V., Kooli, N., and Gravier, G. A Survey on Training and Evaluation of Word Embeddings. *International Journal of Data Science and Analytics*, 11:85–103, February 2021.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. LLaMA: Open and Efficient Foundation Language Models. *arXiv e-prints*, arXiv:2302.13971:1–27, February 2023.

Tsvetkov, Y., Faruqui, M., Ling, W., Lample, G., and Dyer, C. Evaluation of Word Vector Representations by Subspace Alignment. In Màrquez, L., Callison-Burch, C., and Su, J. (eds.), *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP'15)*, pp. 2049–2054, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

Turian, J., Ratinov, L., and Bengio, Y. Word Representations: A Simple and General Method for Semi-Supervised Learning. In Hajič, J. (ed.), *Proceedings of the 48th annual meeting of the association for computational linguistics (ACL'10)*, pp. 384–394, Uppsala, Sweden, July 2010. ACM.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. M., Kaiser, Ł., and Polosukhin, I. Attention is All You Need. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS'17)*, pp. 6000–6010, Long Beach, CA, USA, December 2017. Curran Associates, Inc.

Vilnis, L. and McCallum, A. Word Representations via Gaussian Embedding. *arXiv e-prints*, arXiv:1412.6623: 1–12, December 2014.

Wang, K., Reimers, N., and Gurevych, I. TSDAE: Using Transformer-Based Sequential Denoising Auto-Encoderfor Unsupervised Sentence Embedding Learning. In *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP'21)*, pp. 671–688, Punta Cana, Dominican Republic, November 2021. ACL.

Wang, Y., Wang, K., Gao, F., and Wang, L. Learning Semantic Program Embeddings with Graph Interval Neural Network. In Grove, D. (ed.), *Proceedings of the 35th Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA'20)*, pp. 1–27, Chicago, IL, USA, November 2020. ACM.

Wu, S., Conneau, A., Li, H., Zettlemoyer, L., and Stoyanov, V. Emerging Cross-Lingual Structure in Pretrained Language Models. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL'20)*, pp. 6022–6034. Association for Computational Linguistics, July 2020.

Yin, Z. and Shen, Y. On the Dimensionality of Word Embedding. In Wallach, H., Larochelle, H., Grauman, K., and Cesa-Bianchi, N. (eds.), *Proceedings of the 31st Conference on Neural Information Provessing Systems (NeurIPS'18)*, Montréal, Canada, December 2018. Curran Associates Inc.

Zhang, Z., Fang, M., Chen, L., and Rad, M. R. N. Is Neural Topic Modelling Better than Clustering? An Empirical Study on Clustering with Contextual Embeddings for

Topics. In Carpuat, M., de Marneffe, M.-C., Vladimir, I., and Ruiz, M. (eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'22)*, pp. 3886–3893, Seattle, WA, USA, July 2022. Association for Computational Linguistics.

Zhao, W. X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J.-Y., and Wen, J.-R. A Survey of Large Language Models. *arXiv e-prints*, arXiv:2303.18223: 1–124, November 2023.

Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., and Xu, B. Text Classification Improved by Integrating Bidirectional LSTM with Two-Dimensional Max Pooling. *arXiv e-prints*, arXiv:1611.06639:1–11, November 2016.

Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In Ikeuchi, K., Schnörr, C., Sivic, J., and Vidal, R. (eds.), *Proceedings of the 15th International Conference on Computer Vision (ICCV'15)*, pp. 19–27, Santiago, Chile, December 2015. IEEE.

A. Conditionally Equivalent Output Embeddings

Theorem A.1 (Output Embeddings Equivalence)**.**

1. *If there are* $x_1, \ldots, x_d$ *such that matrix* $\mathcal{B} = [g(x_1), \ldots, g(x_d)]$ *form a basis of* $\mathbb{R}^d$*, and*
2. *there are* $u, v \in \Sigma$ *such that* $u \triangleq v$

then $E^O(u) = E^O(v)$.

Proof. Since our model is optimal, we have that $\forall w \in \Sigma, x \in \mathcal{X} : \; f(w|x) = p(w|x)$ (Here, x can be thought as a context-symbol pair, i.e. $x = (\rho, ?)$). Therefore, since $p(u|x) = p(v|x)$, we have that $f(u|x) = f(v|x)$. Then

$$\forall x : \; \text{softmax}\left(g(x) \cdot E^O\right)_u = \text{softmax}\left(g(x) \cdot E^O\right)_v$$

Using the softmax definition, we have:

$$\forall x : \; \frac{e^{g(x)E^O(u)}}{\sum_j e^{g(x)E^O(j)}} = \frac{e^{g(x)E^O(v)}}{\sum_j e^{g(x)E^O(j)}}$$

Which yield

$$\forall x : \; g(x) \cdot (E^O(u) - E^O(v)) = 0$$

Since this equation holds for all xs, then E^O must satisfy the following system for any $x_1, \ldots, x_d$:

$$\begin{cases} g(x_1) \cdot (E^O(u) - E^O(v)) & = 0 \\ g(x_2) \cdot (E^O(u) - E^O(v)) & = 0 \\ \quad \vdots & \\ g(x_d) \cdot (E^O(u) - E^O(v)) & = 0 \end{cases} \tag{2}$$

Now, when $g(x_1), \ldots, g(x_d)$ form a basis of $\mathbb{R}^d$ the only solution to the previous system becomes $E^O(u) = E^O(v)$. A graphical representation of a 2-dimensional System 2 is depicted in Fig. 3. □

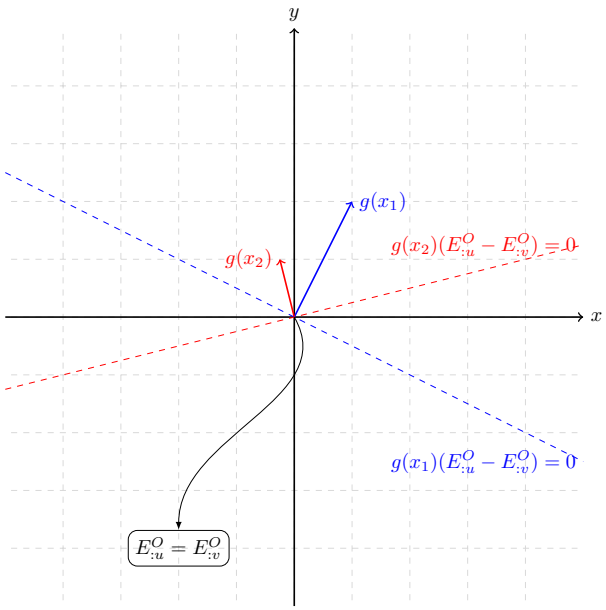


Figure 3. 2-dimensional representation of Proof of Theorem 4.1.

B. Semantically Equivalent Input Embeddings

Theorem B.1 (Input Embeddings Equivalence)**.**

1. *There are symbols $\sigma_1, \ldots, \sigma_d$ such that $\mathcal{B} = E^O(\sigma_1, \ldots, \sigma_d)$ is a basis of $\mathbb{R}^d$, and*
2. *there is a symbol s such that the coefficients a_i of the linear combination $E^O(s) = \sum_i a_i E^O(\sigma_i)$ (such a_i always exists for $\mathcal{B}$ is a basis) do not add up to one, i.e., $\sum_i a_i \neq 1$, and*
3. *there are $u, v \in \Sigma$ such that $u \triangleq v$*

then

$||E^I(u) - E^I(v)|| \leq 2 \min_\rho \left\{ \max \left\{ ||E^I(\rho, u)||, ||E^I(\rho, v)|| \right\} \right\}$

Proof. Firstly, let us recall the NN architecture as the sequence: input embedding layer, self-attention layer, and output embedding layer. In other words,

$$f(X;\theta) = \text{softmax}(g(X;\theta) \cdot E^O) \tag{3}$$
$$g(X;\theta) = \text{softmax}(X_t X^T) X \tag{4}$$
$$X = E^I(\rho, w; \theta) \tag{5}$$

Let t be the index of the mask token, as it is usually done with MLM training. Let us consider the output of the first Layer 5 for a generic sequence:

$$x = (\rho, w) = \underbrace{\sigma_1, \sigma_2, \ldots, \sigma_{t-1}, ?, \sigma_{t+1}, \ldots, \sigma_k}_{\rho}, w, \underbrace{\sigma_{k+2}, \ldots, \sigma_n}_{\rho}$$

Where ρ represents the context around the symbol w. The resulting matrix becomes:

$$E^I(\rho, w) = \begin{bmatrix} \text{———} & E^I(\sigma_1) & \text{———} \\ & \vdots & \\ \text{———} & E^I(?) & \text{———} \\ & \vdots & \\ \text{———} & E^I(w) & \text{———} \\ & \vdots & \\ \text{———} & E^I(\sigma_n) & \text{———} \end{bmatrix} \tag{6}$$

Here, vector $E^I(\sigma_i)$ represents the context symbol embedding of the i-th symbol. $E^I(?)$ is the vector embedding corresponding to the mask token (in the t-th position). $E^I(w)$ represents the *symbol* embedding vector in the context-symbol pair (ρ, w) (in the k-th position). Now, let us refer with X^u and X^v to the matrices representing the context-symbol pairs (ρ, u) and (ρ, v) depicted as follows.

$$E^I(\rho, u) = X^u = \begin{bmatrix} \text{———} & E^I(\sigma_1) & \text{———} \\ & \vdots & \\ \text{———} & E^I(?) & \text{———} \\ & \vdots & \\ \text{———} & E^I(u) & \text{———} \\ & \vdots & \\ \text{———} & E^I(\sigma_n) & \text{———} \end{bmatrix} \tag{7}$$

$$E^I(\rho, v) = X^v = \begin{bmatrix} \text{———} & E^I(\sigma_1) & \text{———} \\ & \vdots & \\ \text{———} & E^I(?) & \text{———} \\ & \vdots & \\ \text{———} & E^I(v) & \text{———} \\ & \vdots & \\ \text{———} & E^I(\sigma_n) & \text{———} \end{bmatrix} \tag{8}$$

Note that these matrices differ only on k-th row, where in X^u we have $E^I(u)$ and in X^v we have $E^I(v)$. Under semantic equivalence assumption and optimality assumption, we have that:

$$f(X^u) = f(X^v) \tag{9}$$
$$\implies \text{softmax}(g(X^u;\theta)\cdot E^O) = \text{softmax}(g(X^v;\theta)\cdot E^O) \tag{10}$$

Using the softmax definition, it is easy to show that $\text{softmax}(a) = \text{softmax}(b) \implies \exists \vec{c}.\, a = b + \vec{c}$. Where $\vec{c}$ is the constant vector $\vec{c} = [c, \dots, c]$ Briefly:

$$\text{softmax}(a) = \text{softmax}(b) \tag{11}$$
$$\implies \forall i:\ \text{softmax}(a)_i = \text{softmax}(b)_i \tag{12}$$
$$\implies \forall i:\ \frac{e^{a_i}}{\sum_j e^{a_j}} = \frac{e^{b_i}}{\sum_j e^{b_j}} \tag{13}$$
$$\implies \forall i:\ a_i = b_i + \underbrace{\ln(\frac{\sum_j e^{b_j}}{\sum_j e^{a_j}})}_{c} \tag{14}$$
$$\implies \exists \vec{c}.\, a = b + \vec{c} \tag{15}$$

Therefore, using the previous implication we can derive:

$$\text{softmax}(g(X^u;\theta)\cdot E^O) = \text{softmax}(g(X^v;\theta)\cdot E^O) \tag{16}$$
$$\implies \exists \vec{c}.\, g(X^u;\theta)\cdot E^O = g(X^v;\theta)\cdot E^O + \vec{c} \tag{17}$$
$$\implies \exists \vec{c}.\, (g(X^u;\theta) - g(X^v;\theta))\cdot E^O = \vec{c} \tag{18}$$

Using the first two hypothesis of the theorem, it is fairly easy to show that $\vec{c} = \vec{0}$. Briefly, let $\vec{g} = (g(X^u;\theta) - g(X^v;\theta))$ and $\vec{s} = E^O(s)$. We have that $\vec{g}\vec{s} = c$. However, we can express the vector $\vec{s}$ as a linear combination of the basis vector $\vec{b_i}$ in $\mathcal{B}$, i.e. $\vec{s} = \sum_i a_i \vec{b_i}$. However, $\vec{g}\vec{b_i} = c$, therefore:

$$\vec{g}\vec{s} = c \implies \sum_i a_i \underbrace{\vec{g}\vec{b_i}}_{c} = c \implies \sum_i a_i = 1$$

Which yield a contradiction of the second hypothesis in the theorem definition unless $c = 0$.Therefore, we obtain:

$$(g(X^u;\theta) - g(X^v;\theta))E^O = \vec{0} \tag{19}$$

Further, since E^O contains a basis we also have that the only solution to previous equation is $g(X^u;\theta) = g(X^v;\theta)$ yielding:

$$g(X^u;\theta) - g(X^v;\theta) = \vec{0} \tag{20}$$
$$\implies \underbrace{\text{softmax}(X^u_t (X^u)^T)}_{A^u_t} X^u - \underbrace{\text{softmax}(X^v_t (X^v)^T)}_{A^v_t} X^v = \vec{0} \tag{21}$$
$$\implies A^u_t X^u - A^v_t X^v = \vec{0} \tag{22}$$

Now, from this equation, we proceed with two different parallel derivations:

$$A_t^u X^u - A_t^v X^v = \vec{0} \quad (23)$$
$$\implies A_t^u X^u + A_t^v X^u - A_t^v X^u - A_t^v X^v = \vec{0} \quad (24)$$
$$\implies A_t^v (X^u - X^v) + (A_t^u - A_t^v) X^u = \vec{0} \quad (25)$$
$$\implies A_{tk}^v (X_k^u - X_k^v) + (A_t^u - A_t^v) X^u = \vec{0} \quad (26)$$
$$\implies A_{tk}^v (E^I(u) - E^I(v)) + (A_t^u - A_t^v) X^u = \vec{0} \quad (27)$$
$$\implies (E^I(u) - E^I(v)) = \frac{(A_t^v - A_t^u) X^u}{A_{tk}^v} \quad (28)$$
$$\implies ||E^I(u) - E^I(v)|| \leq \frac{||A_t^v - A_t^u|| ||X^u||}{A_{tk}^v} \quad (29)$$

$$A_t^u X^u - A_t^v X^v = \vec{0} \quad (30)$$
$$\implies A_t^u X^u - A_t^v X^v - A_t^u X^v + A_t^u X^v = \vec{0} \quad (31)$$
$$\implies A_t^u (X^u - X^v) + (A_t^u - A_t^v) X^v = \vec{0} \quad (32)$$
$$\implies A_{tk}^u (X_k^u - X_k^v) + (A_t^u - A_t^v) X^v = \vec{0} \quad (33)$$
$$\implies A_{tk}^u (E^I(u) - E^I(v)) + (A_t^u - A_t^v) X^v = \vec{0} \quad (34)$$
$$\implies (E^I(u) - E^I(v)) = \frac{(A_t^v - A_t^u) X^v}{A_{tk}^u} \quad (35)$$
$$\implies ||E^I(u) - E^I(v)|| \leq \frac{||A_t^v - A_t^u|| ||X^v||}{A_{tk}^u} \quad (36)$$

Here, Equations 26 and 33 follow from the respective previous by the fact that X^u and X^v differ only in the k-th row (the one corresponding to embeddings of the semantically equivalent symbols u and v). These two derivations are necessary to produce tight bounds for the quantity $||E^I(u) - E^I(v)||$.

$$||E^I(u) - E^I(v)|| \leq \min \left\{ \frac{||A_t^v - A_t^u|| ||X^u||}{A_{tk}^v}, \frac{||A_t^v - A_t^u|| ||X^v||}{A_{tk}^u} \right\} \quad (37)$$
$$\leq \frac{||A_t^v - A_t^u||}{\max\{A_{tk}^u, A_{tk}^v\}} \max\{||X^u||, ||X^v||\} \quad (38)$$
$$\leq \sqrt{\sum_i \frac{(A_{ti}^v - A_{ti}^u)^2}{\max\{A_{tk}^u, A_{tk}^v\}^2}} \max\{||X^u||, ||X^v||\} \quad (39)$$
$$\leq \sqrt{1 + \sum_{i \neq k} \frac{(A_{ti}^v - A_{ti}^u)^2}{\max\{A_{tk}^u, A_{tk}^v\}^2}} \max\{||X^u||, ||X^v||\} \quad (40)$$

Now recall that:

$$A_{ti}^w = \text{softmax}(X_t^w (X^w)^T)_i = \frac{\exp(X_t^w X_{:i}^w)}{\sum_j \exp(X_t^w X_{:j}^w)} = \frac{\exp(X_t^w X_{:i}^w)}{\exp(X_t^w X_{:i}^w) + \sum_{j \neq i} \exp(X_t^w X_{:j}^w)} \quad (41)$$

Note that $X_i^u = X_i^v$ when $i \neq k$ (X^u and X^v differ only in the k-th row where the first has the vector $E^I(u)$ and the latter has $E^I(v)$). In this case, the exponential sum $\sum_{j \neq k} \exp(X_t^w X_{:j}^w)$ is the same in the case of A^u and A^v, so let us call this sum β. For simplicity, when $i \neq k$ let us refer to $\exp(X_t^u X_{:i}^u) = \exp(X_t^v X_{:i}^v) = \alpha_i$. Instead, when $i = k$ we will refer to $\exp(X_t^u X_{:i}^u) = \alpha_u$ and $\exp(X_t^v X_{:i}^v) = \alpha_v$. In other words:

$$i \neq k \implies A_{ti}^u = \frac{\alpha_i}{\beta + \alpha_u} \qquad i \neq k \implies A_{ti}^v = \frac{\alpha_i}{\beta + \alpha_v} \qquad i = k \implies A_{ti}^u = \frac{\alpha_u}{\beta + \alpha_u} \qquad i = k \implies A_{ti}^v = \frac{\alpha_v}{\beta + \alpha_v}$$

Now, we continue from the previous bounding:

$$= \sqrt{1 + \sum_{i \neq k} \frac{(A^v_{ti} - A^u_{ti})^2}{\max\{A^u_{tk}, A^v_{tk}\}^2}} \max\{||X^u||, ||X^v||\} \tag{42}$$

$$= \sqrt{1 + \sum_{i \neq k} \frac{(\frac{\alpha_i}{\beta+\alpha_v} - \frac{\alpha_i}{\beta+\alpha_u})^2}{\max\{A^u_{tk}, A^v_{tk}\}^2}} \max\{||X^u||, ||X^v||\} \tag{43}$$

$$= \sqrt{1 + \left(\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right)^2 \sum_{i \neq k} \frac{\alpha_i^2}{\max\{A^u_{tk}, A^v_{tk}\}^2}} \max\{||X^u||, ||X^v||\} \tag{44}$$

$$\leq \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{1}{\max\{A^u_{tk}, A^v_{tk}\}} \sqrt{\sum_{i \neq k} \alpha_i^2}\right) \max\{||X^u||, ||X^v||\} \tag{45}$$

$$\leq \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{1}{\max\{A^u_{tk}, A^v_{tk}\}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{46}$$

$$= \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{1}{\max\left\{\frac{\alpha_u}{\beta+\alpha_u}, \frac{\alpha_v}{\beta+\alpha_v}\right\}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{47}$$

$$= \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{1}{\max\left\{1 - \frac{\beta}{\beta+\alpha_u}, 1 - \frac{\beta}{\beta+\alpha_v}\right\}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{48}$$

$$= \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{1}{1 - \beta\frac{1}{\max\{\beta+\alpha_u, \beta+\alpha_v\}}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{49}$$

$$= \left(1 + \left|\frac{1}{\beta+\alpha_v} - \frac{1}{\beta+\alpha_u}\right| \frac{\beta + \max\{\alpha_u, \alpha_v\}}{\max\{\alpha_u, \alpha_v\}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{50}$$

$$= \left(1 + \left|\frac{\alpha_v - \alpha_u}{(\beta+\alpha_v)(\beta+\alpha_u)}\right| \frac{\beta + \max\{\alpha_u, \alpha_v\}}{\max\{\alpha_u, \alpha_v\}} \beta\right) \max\{||X^u||, ||X^v||\} \tag{51}$$

$$\leq \left(1 + \frac{|\alpha_v - \alpha_u|}{\max\{\alpha_u, \alpha_v\}}\right) \max\{||X^u||, ||X^v||\} \tag{52}$$

$$\leq 2 \max\{||X^u||, ||X^v||\} \tag{53}$$

$$\tag{54}$$

Therefore, we obtained:

$$||E^I(u) - E^I(v)|| \leq 2 \max\{||X^u||, ||X^v||\} \tag{55}$$

Furthermore, this inequality must hold for any possible context ρ in which symbols u and v could be found. Therefore, we can update the bound as follows:

$$||E^I(u) - E^I(v)|| \leq 2 \min_{\rho} \left\{\max\left\{||E^I(\rho, u)||, ||E^I(\rho, v)||\right\}\right\} \tag{56}$$

Which concludes the proof.

□

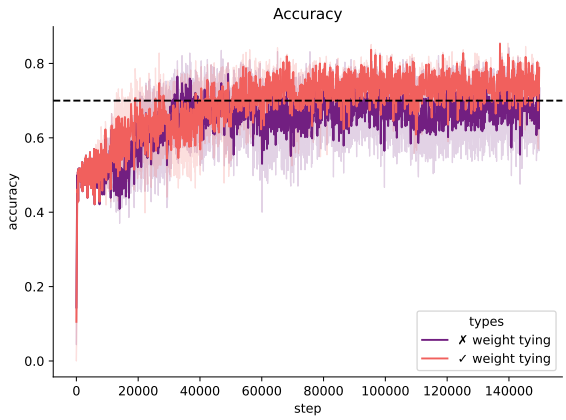
C. Ablation Study

C.1. What happens when the distribution hypothesis does hold?

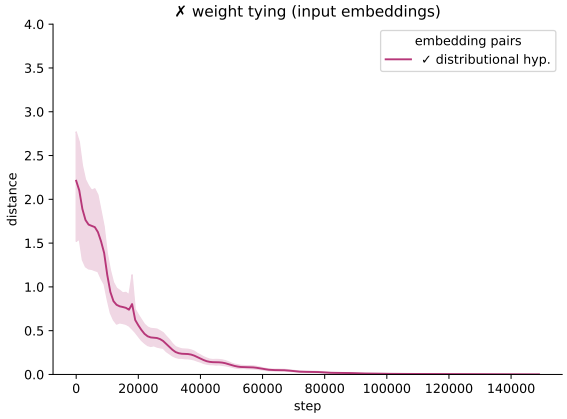
Experiment. In Sect. 5, we exclusively examined a scenario in which the distributional hypothesis held for one pair of symbols (1_A and 1_B) but did not for the other pair (0_A and 0_B). Here, we consider the EXor problem where the distributional hypothesis holds for all symbols. This can be achieved by simply removing the problematic pair (0_A and 0_B).

Expectations. In this scenario, we anticipate that a tied model would exhibit slightly faster results compared to an untied one. This expectation arises from both the input and output embedding matrices of the tied model attempting to encode the same semantic information in the embeddings. Furthermore, we predict that the embeddings of 1_A and 1_B will converge, becoming closer to each other, given their semantic and conditional equivalence. Specifically, the tied model is expected to converge faster wrt. the untied model.

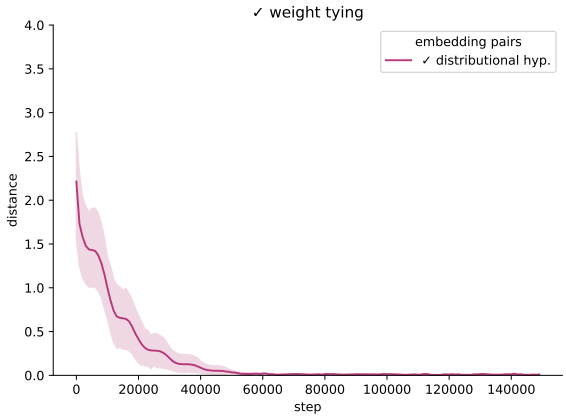
Result. Fig. 4 shows the results. As one would expect, in this scenario, the tied model works slightly better in terms of accuracy. Also, the untied model does encode 1_A and 1_B close to each other in both the input and output embedding layer since $1_A \stackrel{\circ}{=} 1_B$ and $1_A \triangleq 1_B$. However, these embeddings become close to each other faster in the tied model wrt. the untied model.



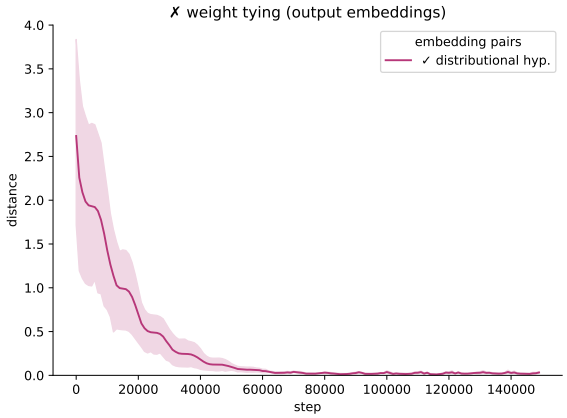
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) 1_A and 1_B distance in E^I layer of the untied model



(c) 1_A and 1_B distance in $E^O = E^I$ layer of the untied model



(d) 1_A and 1_B distance in E^O layer of the untied model

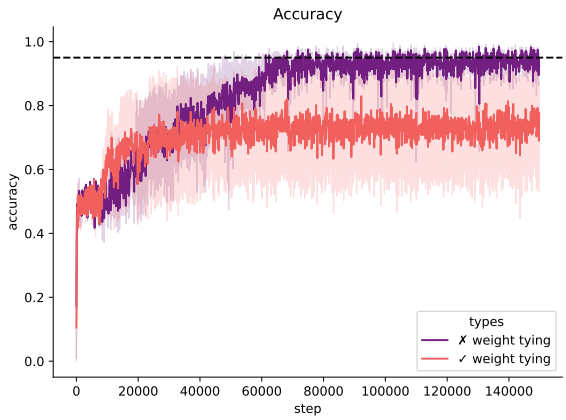
Figure 4. Accuracy and embedding distances for the tied and untied model when the distributional hypothesis hold for all symbols and there are semantically and conditionally equivalent symbols

C.2. What happens when the distribution hypothesis does not hold?

Experiment. In Sect. 5, we exclusively examined a scenario in which the distributional hypothesis held for one pair of symbols ($\texttt{1}_\texttt{A}$ & $\texttt{1}_\texttt{B}$) but did not for the other pair ($\texttt{0}_\texttt{A}$ & $\texttt{0}_\texttt{B}$). Here, we consider the EXor problem where the distributional hypothesis does not hold for all symbols. This can be achieved by simply removing the semantically and conditionally equivalent pair ($\texttt{1}_\texttt{A}$ & $\texttt{1}_\texttt{B}$).

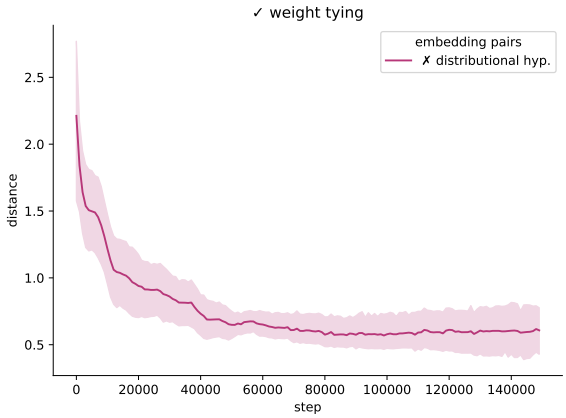
Expectation. In this scenario, we expect that an untied model should achieve results slightly better wrt. a tied one as both input and output embedding matrices try to encode different semantic information in the same embeddings. Additionally, we also expect embeddings of $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$ to be distanced from each other in the output matrix (as these symbols are not conditionally equivalent).

Results. Fig. 5 shows the results. In this scenario, the untied model works much better in terms of accuracy. Note that, while the untied model can catch up with the tied model when all symbols respect the distributional hypothesis, the vice-versa is not true. This is because the tied model can always reach a parameter configuration such that $E^I(u) = E^O(v)$ when $u \stackrel{\circ}{=} v \iff u \triangleq v$. On the other hand, when $u \stackrel{\circ}{=} v \not\iff u \triangleq v$ then $E^I(u)$ should be different from $E^O(v)$ but this cannot happen if we tie their weights. Also, the untied model does encode $\texttt{1}_\texttt{A}$ and $\texttt{1}_\texttt{B}$ close to each other in the input and far apart from each other in the output embedding layer since $\texttt{1}_\texttt{A} \stackrel{\circ}{=} \texttt{1}_\texttt{B}$ and $\texttt{1}_\texttt{A} \not\triangleq \texttt{1}_\texttt{B}$, as one would expect. On the other hand, the tied model encodes $\texttt{1}_\texttt{A}$ far from $\texttt{1}_\texttt{B}$ while they should be close to each other in the input embedding layer and far in the output embedding layer.

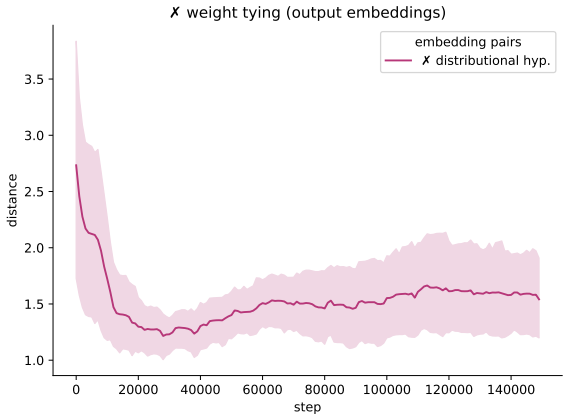


(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)

(b) $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$ distance in E^I layer of the untied model



(c) $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$ distance in $E^O = E^I$ layer of the untied model



(d) $\texttt{0}_\texttt{A}$ and $\texttt{0}_\texttt{B}$ distance in E^O layer of the untied model

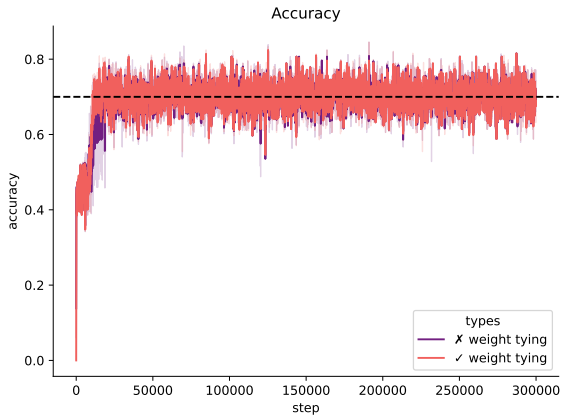
Figure 5. Accuracy and embedding distances for the tied and untied model when there are not symbols both semantically and conditionally equivalent.

C.3. What happens when the transformer model is larger?

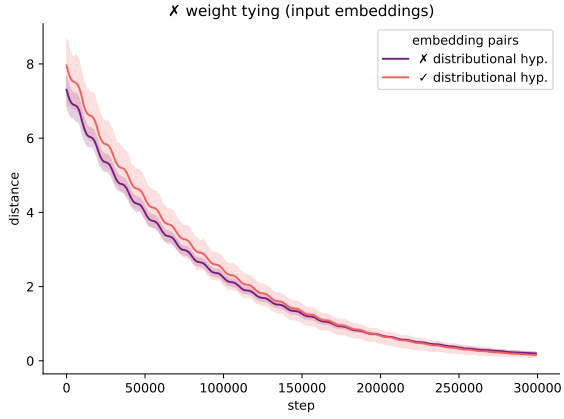
Experiment. In Sect. 5, we exclusively examined a scenario with an extremely small architecture. Here, we employ a slightly larger architecture that is over-parametrized for the problem at hand. In particular, we use a 2-layer (previously 1-layer) transformer architecture, with an input and output embedding size of 32 (previously 4).

Expectation. We expect to observe a behavior similar to those observed in the main experiment (Sect. 5).

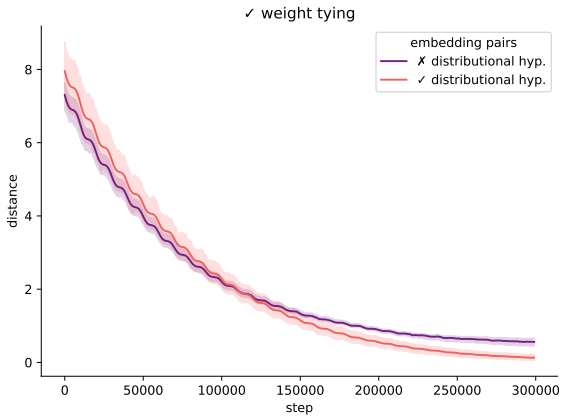
Results. Surprisingly, while we would expect the untied model to be slightly more accurate this does not happen. We believe that the tied model is able to compensate (the fact of having input and output embedding matrices tied) by using the additional parameters. Moreover, the tied model is not particularly fast in encoding $\texttt{1}_{\texttt{A}}$ close to $\texttt{1}_{\texttt{B}}$ wrt. tied model. We believe that over-parametrization is the cause of this behavior, as achieving good results early alleviates the pressure on having properly aligned embeddings. Notable, we can still observe that $\texttt{1}_{\texttt{A}}$ and $\texttt{1}_{\texttt{B}}$ (which are both semantically and conditionally equivalent) converge. Meanwhile, $\texttt{0}_{\texttt{A}}$ and $\texttt{0}_{\texttt{B}}$ (which are not conditionally equivalent) become close to each other only in the input embedding layer of the untied model.



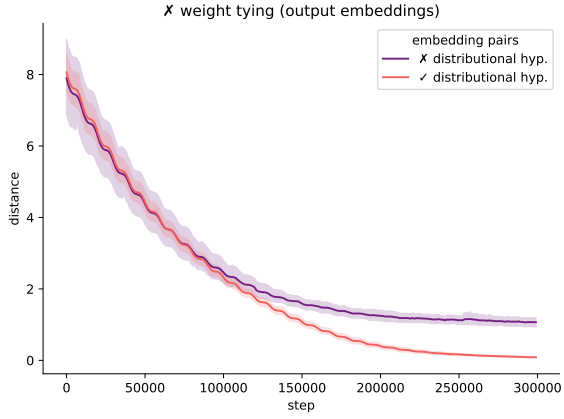
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) E^I distances for the untied model



(c) Embedding distances for the tied model



(d) E^O distances for the untied model

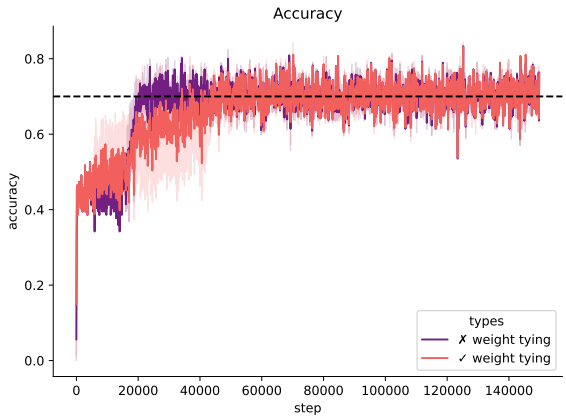
Figure 6. Accuracy and embedding distances for the tied and untied model when the transformer is slightly larger

C.4. What happens when we use and LSTM architecture?

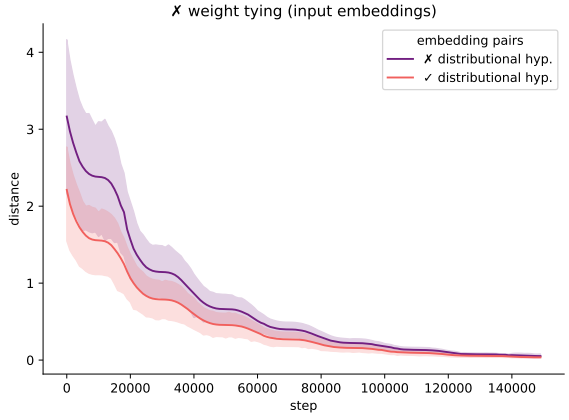
Experiment. In Sect. 5, we exclusively examined a scenario with a transformer architecture. Here, we employ a different architecture. In particular, we use a 2-layer (a 1-layer architecture was not able to generalize) bidirectional LSTM (Hochreiter & Schmidhuber, 1997) architecture (we use the PyTorch implementation[5]), with input and output embedding size of 4.

Expectation. We expect to observe a behavior similar to those observed in the main experiment (Sect. 5).

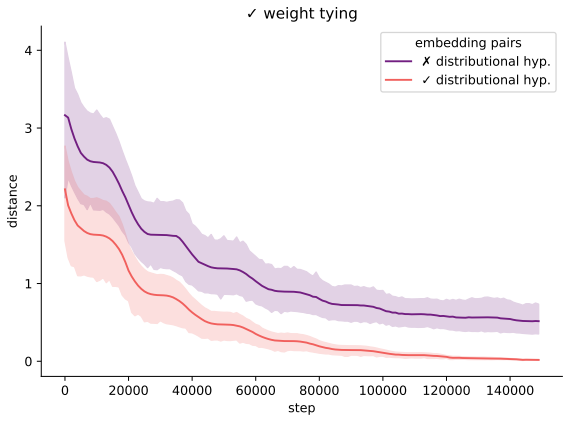
Results. Mostly, in Fig. 7, we can observe a behavior similar to the one already observed for the transformer architecture. However, the rate at which embedding becomes close to each other is a bit slower wrt. the transformer architecture. This may be the result of slight over-parametrization or it may be the case that the bound provided in Theorem 4.2 is less tight.



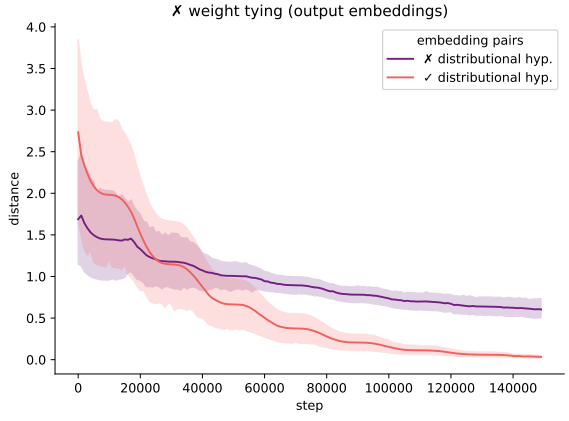
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) E^I distances for the untied model



(c) Embedding distances for the tied model



(d) E^O distances for the untied model

Figure 7. Accuracy and embedding distances for the tied and untied model when the model is a LSTM

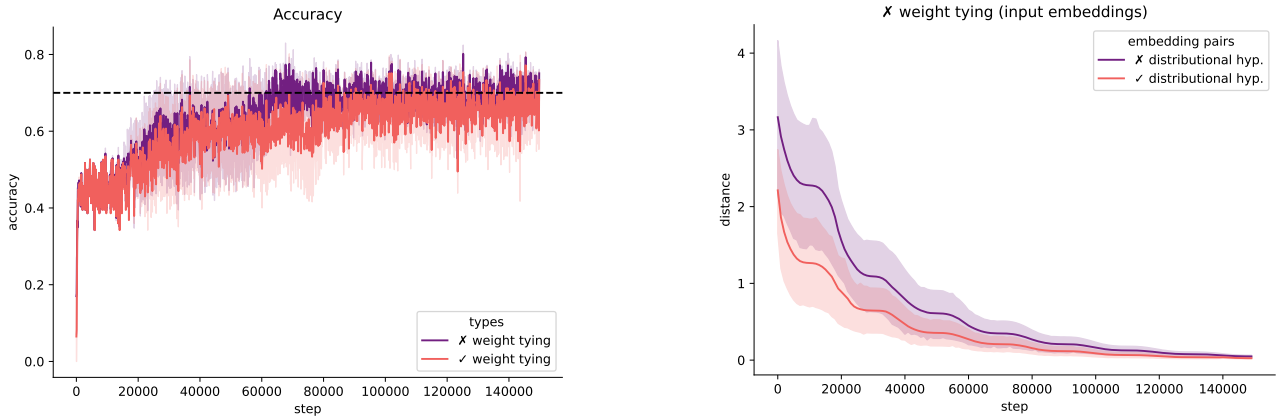
[5] https://pytorch.org/docs/stable/generated/torch.nn.LSTM.html

C.5. What happens when we use and MLPMixer architecture?

Experiment. Until now, we have examined a scenario with an attention-based architecture (the Transformer), and a recurrent neural network (the LSTM). Here we introduce multi-layer-perceptron-based architecture—the MLPMixer (Tolstikhin et al., 2021). In particular, we use a 1-layer MLPMixer architecture implemented from scratch, with an input and output embedding size of 4.

Expectation. We expect to observe a behavior similar to those observed in the main experiment (Sect. 5).

Result. Mostly, in Fig. 8, we can observe a behavior similar to the one already observed for the transformer architecture. The rate at which embedding becomes close to each other is bit slower wrt. the transformer architecture but faster wrt. the LSTM architecture. Again, we believe that this is the result of either a slight over-parametrization or a less tight bound for Theorem 4.2.



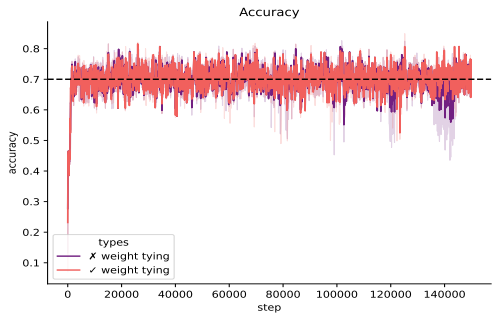
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)

(b) E^I distances for the untied model

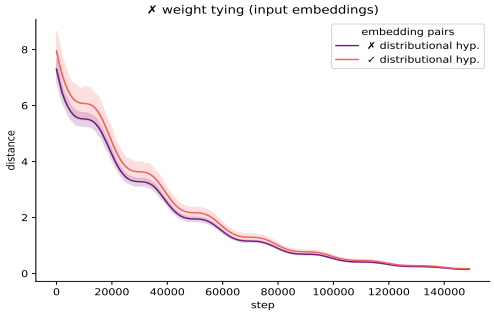
(c) Embedding distances for the tied model

(d) E^O distances for the untied model

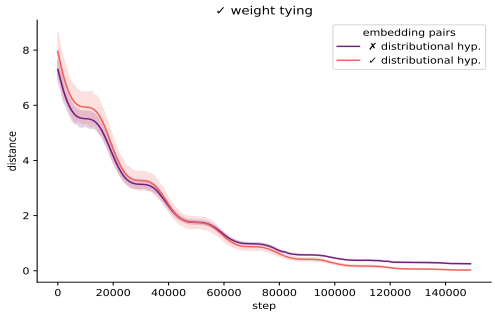
Figure 8. Accuracy and embedding distances for the tied and untied model when the model is a MLPMixer



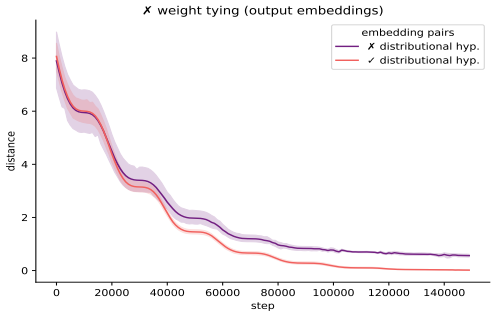
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) 1_A and 1_B distance in E^I layer of the untied model

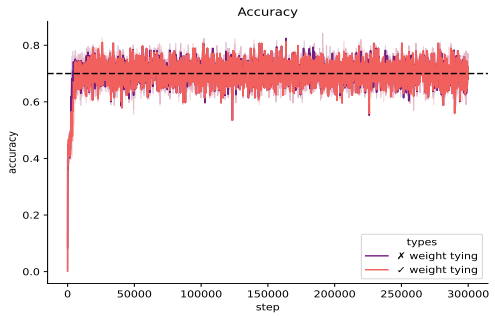


(c) 1_A and 1_B distance in $E^O = E^I$ layer of the untied model

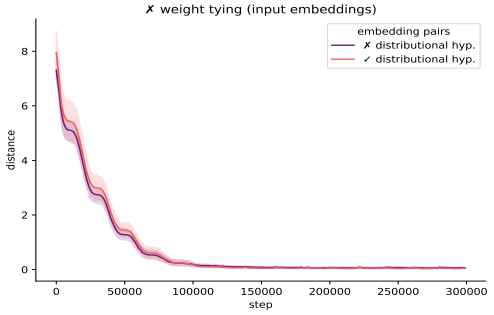


(d) 1_A and 1_B distance in E^O layer of the untied model

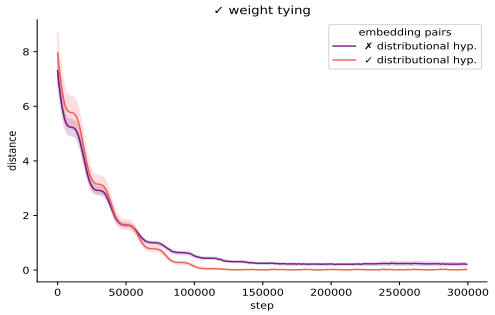
Figure 9. Embedding distances for a larger LSTM.



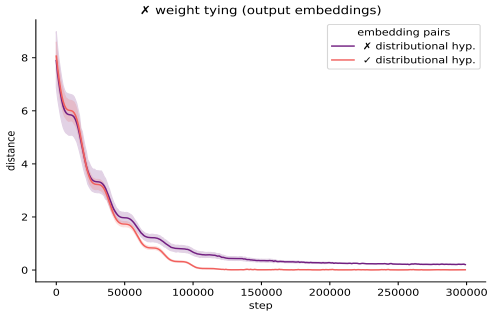
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) 1_A and 1_B distance in E^I layer of the untied model



(c) 1_A and 1_B distance in $E^O = E^I$ layer of the untied model



(d) 1_A and 1_B distance in E^O layer of the untied model

Figure 10. Embedding distances for a larger MLPMixer.

C.6. Bigger LSTM & bigger MLPMixer

For completeness, we also provide the embedding distance for a larger version of an LSTM and MLPMixer architecture. However, these figures (Fig. 9 and Fig. 10) simply confirm the results already observed in the previous scenarios.

The LSTM architecture is a 4-layer bidirectional LSTM architecture with input and output embedding size of 32 parameters. The MLPMixer is a 2-layer MLPMixer architecture with an input and output embedding size of 32. The remaining parameters remain unchanged.

C.7. What happens when we use natural data?

Overview. In Section 5, we focused solely on a scenario involving an artificially generated small dataset. Here, we aim to expand the experiment to a larger and more natural dataset to assess the extent to which Theorems 4.1 and 4.2 hold.

Dataset. We chose the Bookcorpus dataset (Zhu et al., 2015), a 5GB collection of English sentences extracted from existing books. Each sentence was tokenized using a pre-trained tokenizer (Devlin et al., 2019) from the Huggingface dataset library. [6] Additionally, we curated a set of 100 common tokens, denoted as C (e.g., "the", "time", "two"), to serve as references. For each token in C, we assigned two symbols representing the same token (e.g., "the" is represented by symbols $1,996$ and $30,520$). During training, each token in C was randomly replaced with equal probability by either one of the two symbols (e.g., "the" could be replaced with probability $1/2$ by either $1,996$ or $30,520$). Consequently, we established 100 symbol pairs for which the distributional hypothesis holds.

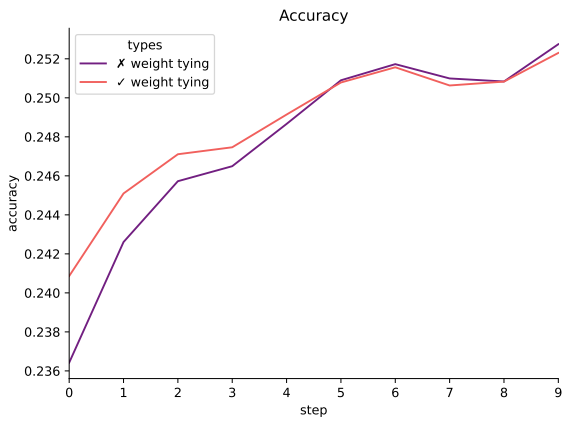
Task. We mask a token in a sequence with a probability of 0.15, and the model's objective is to predict the masked tokens based on the remaining context.

Model. We adopt a traditional 3-layer transformer architecture with the following hyperparameters: 4 attention heads, 128 embedding size, 512 feed-forward size, and gelu activation. Additionally, we initialize embeddings from a normal distribution with mean 0 and standard deviation 0.3. The model is trained for 10 epochs using the AdamW optimizer with a learning rate of $1\mathrm{e}{-4}$ and weight decay of $1\mathrm{e}{-2}$.

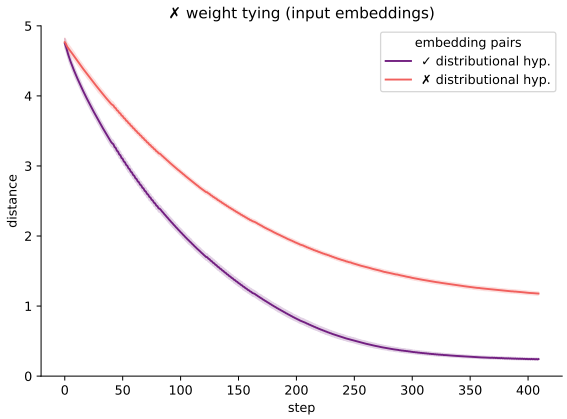
Expectation. Note that symbols representing the same token in C are semantically equivalent. Consequently, during training, we expect the respective input embeddings to become close to each other (according to Theorem 4.2). Similarly, these symbols are also conditionally equivalent, suggesting that the respective output embeddings should likewise converge (according to Theorem 4.1). Conversely, when comparing symbols representing different tokens in C, they are neither semantically nor conditionally equivalent. Hence, their input and output embeddings should keep their distance from each other.

Result. The results are depicted in Fig. 11. Firstly, let us discuss the untied model. Both Fig.11b and Fig.11d demonstrate the expected behavior. Here, semantically equivalent symbols are encoded close to each other in the input embedding matrix, and similarly, conditionally equivalent symbols are encoded close to each other in the output embedding matrix. This trend is also observed in the case of the tied model, as shown in Fig.11c. Finally, Fig. 11a presents the validation accuracy during the 10 training epochs. It can be observed that the tied model initially outperforms the untied model, but the untied model catches up later during training.

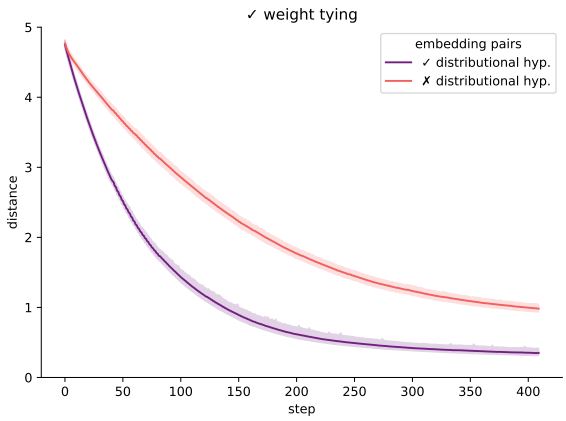
[6] https://huggingface.co/google-bert/bert-base-uncased



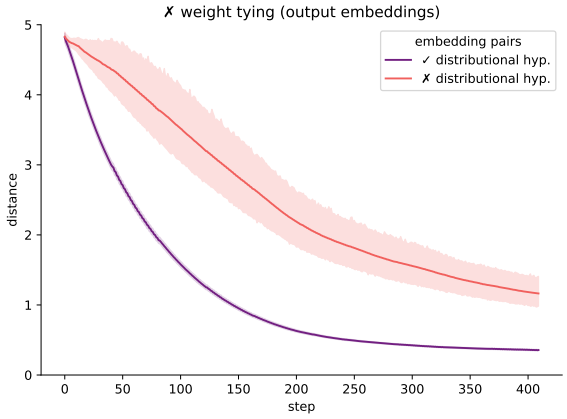
(a) Tied (✓ W.T.) vs. the Untied model (✗ W.T.)



(b) E^I distances for the untied model



(c) Embedding distances for the tied model



(d) E^O distances for the untied model

Figure 11. Accuracy and embedding distances for the tied and untied model for the bookcorpus dataset