BANYAN: IMPROVED REPRESENTATION LEARNING WITH EXPLICIT STRUCTURE

Anonymous authors
Paper under double-blind review

ABSTRACT

We present Banyan, a model that efficiently learns semantic representations by leveraging an inductive bias towards explicit hierarchical structure. Although typical transformer-based models excel at scale, they struggle in low-resource settings. Recent work on models exploiting explicit structure has shown promise as efficient learners in resource-constrained environments. However, these models have yet to demonstrate truly competitive performance. Banyan bridges this gap, significantly improving upon prior structured models and providing, for the first time, a viable alternative to transformer embeddings for under-represented languages. We achieve these improvements through two key innovations 1) A novel entangled tree structure that resolves multiple constituent structures into a single shared one, explicitly incorporating global context. 2) Diagonalized message passing functions that increase the influence of the inductive bias. Our final model has just 14 non-embedding parameters yet is competitive with baselines many orders of magnitude larger. Banyan outperforms its structured predecessors and competes with large unstructured models across various semantic tasks in multiple languages. Notably, it excels in low-resource settings, highlighting its potential for efficient and interpretable NLP in resource-constrained environments. These results underscore the value of appropriate inductive biases in capturing semantic relationships and open new avenues for efficient, interpretable NLP models.

1 INTRODUCTION

Semantic representations of text are important for many NLP applications such as retrieval augmented generation (Lewis et al., 2020), question answering, and summarisation (Abdalla et al., 2023; Wang et al., 2022). They are also useful for clustering and organising textual data when labelled training sets are not available. At the time of writing such representations are primarily generated by large scale transformer models (Vaswani et al., 2017). These models are incredibly effective, but training them usually requires scale, both in terms of data and compute.

An alternative approach is to take inspiration from linguistics/cognitive science and explicitly incorporate structured compositions. Put simply, composition states that all you need to understand the semantics of a whole are the meanings of its parts and the structure that dictates how they fit together (Chomsky, 1956; Crain & Nakayama, 1987; Pallier et al., 2011; de Marneffe et al., 2006). This is a very efficient principle, because novel utterances can broken down into familiar parts using systematic rules, rather than having to store the meaning of each utterance individually. It is thought that this principle lets humans generalise from (comparatively) little data and makes us efficient learners (Fodor & Pylyshyn, 1988; Lake et al., 2016; Ito et al., 2022; Wiedemer et al., 2023). In order to explicitly incorporate an inductive bias of this kind we need to change the modelling process somewhat. Rather than keeping all the information flow internal, models must now learn representations for the atomics, operate on (and/or learn) a discrete graph that dictates the mode of combination, and learn functions that control information flow through such a graph. Models of this kind have demonstrated improved language modelling perplexity at cognitively plausible scales (Hu et al., 2021; 2022); better systematic generalisation (Sartran et al., 2022; Murty et al., 2023); and, importantly for this paper, the ability to efficiently acquire semantics (Opper et al., 2023).

Opper et al. (2023) introduce a model called the `Self-StrAE`, which learns to representations which have to explicitly model compositional semantics. This demonstrated very promising performance

while requiring minimal resources. Both in terms of data and model size. Opening the door to investigate whether more compute efficient solutions can be found for learning semantic representations. This would be particularly useful for low resourced languages where relying on scale is not a generally feasible solution. However, the `Self-StrAE`, while promising, still lags behind large scale pre-trained transformers, even in langauges which fall outside of standard pre-training corpora. In this paper we introduce a model called `Banyan`, which significantly improves performance over that of (Opper et al., 2023), while simultaneously achieving even greater resource efficiency. We achieve this by changing the form of the structure optimised to a graph that models global relations between nodes which we call an entangled trees, as well as a message passing regime based on diagonal functions which reduces parameters while producing more expressive representations. Our model, `Banyan`, achieves competitive performance with transformer based baselines, and for the first time represents a low cost yet viable alternative for producing representations for low resource languages, measured using semantic textual similarity (STS) tasks. By leveraging cognitively inspired inductive biases we can achieve performance comparable or better than large scale pre-trained LLMs but with only 14 non-embedding parameters.

2 Background and Related Work

Banyan is a graph neural network, specifically a recursive neural network (RvNN), that learns both structure and representations. Before detailing the model, we unpack these terms in this section.

Recursive Neural Networks: Like their recurrent cousins, recursive neural networks operate by repeatedly applying a function to update a the network state in an ordered fashion. However, rather than utilising temporal ordering (i.e. over a sequence), RvNNs operate according to some hierarchical structure, typically this given as input and most often it is a binary tree. They can be applied bottom-up (traversing from leaves to root) or top-down (from root to leaves) or both. First popularised by Socher et al. (2011; 2013), they have inspired numerous successor frameworks which differ in how the recursive function is defined. These successors include the Tree-LSTM (Tai et al., 2015), IORNN (Le & Zuidema, 2014; Ji & Eisenstein, 2015) and also `Banyan`.

Learning Structure: RvNNs typically require structure as input, sometimes such structure is available or can be obtained using existing tools, but generally this is quite a limiting factor, because it limits model flexibility. A solution is to incorporate a mechanism within the model that is able to induce the structure during the recursive computation. Prior approaches include the use of differentiable chart parsing (Drozdov et al., 2019; 2020; Hu et al., 2021; 2022), beam search (Ray Chowdhury & Caragea, 2023), continuous relaxation (Chowdhury & Caragea, 2021; Soulos et al., 2024), or reinforcement learning (Havrylov et al., 2019). While successful these solutions can suffer from memory issues and hyperparameter sensitivity. In this paper we adopt the approach of Opper et al. (2023) and utilise representation similarity to dictate merge order. This is both computationally inexpensive and surprisingly effective.

Semantic Representations of Text: Systems like Word2Vec (Mikolov et al., 2013) and GLoVe (Pennington et al., 2014) model the semantics of words using the distributional hypothesis (Harris, 1954). This hypothesis states that the context a word is used in defines its meaning. Consequently, representations are learned by setting a context window of some fixed size and then using that to predict the missing word. This approach proved very effective for a long time, but words don't all have one meaning - it changes in context. A natural solution is to use transformers. Initially smaller (relative to today) encoder only models produced poor representations (Reimers & Gurevych, 2019). However, at time of writing transformers with optional contrastive finetuning (Gao et al., 2021) have become the model of choice for producing semantic representations.

Semantic Representation Learning through Structure: Transformer embeddings are more successful than static word embeddings because they are allow for flexible contextualisation. The meaning of a word can change in context and will more strongly influenced by certain neighbours rather than others. A transformer can model this phenomenon by routing information between specific tokens via its attention. An alternative approach is to use structure, where rather than attention dictating routing, it is done by an explicit graph. Early work in this area focused on lexical semantics, using dependency parses to determine a more focused context window (Levy & Goldberg, 2014; Vashishth et al., 2019). More recent work by Opper et al. (2023) use constituency parses in order to learn embeddings at both the word and the sentence level. They introduce two variants of their model. StrAE: which takes

structure as input, and Self-StrAE: which learns its own structure with the representations. This later model, the `Self-StrAE`, is the starting point from which we build `Banyan`, and will be outlined in more detail in the subsequent section.

3 PRELIMINARY: SELF-STRAE

`Self-StrAE` involves three main components that acts over a sentence $\mathbf{w} = \langle w_n \rangle_{n=1}^{N}$ represented as a sequence of tokens. These are: (a) a procedure to determine which tokens to merge and in what order, (b) message passing functions—composition and decomposition—that merge and split embeddings respectively, and (c) a reconstructive objective that leverages both the induced structure and embeddings. While we refer the reader to Opper et al. (2023) for a detailed description of these components, we briefly recap its operation to provide sufficient background for the development of `Banyan` (§ 4).

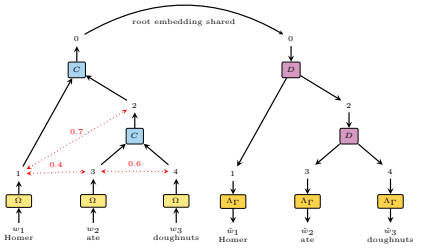


Figure 1: Self-StrAE operation. Red lines indicate cosine similarity. Shared colours imply shared parameters.

At a high-level, `Self-StrAE` learns representations that both define their own structure and are in turn defined by it. This is achieved by first *embedding* tokens to form an initial frontier using an embedding matrix Ω_Ψ. Next it then takes the adjacent tokens with the highest cosine similarity to each other (*ate* and *doughnuts* in Figure 1) and merges them into a single embedding using a parametric *composition function* C_Φ. This procedure is repeated until the sequence reduces to a single root embedding. The resulting merge history is then treated as the induced binary tree for the sentence. `Self-StrAE` then traverses back down the structure, recursively splitting embeddings at every node using a parametric *decomposition function* D_Θ to recover embeddings for the leaves. Finally, the model can optionally use a *dembedding function* Λ_Γ to predict tokens $\widehat{w}_n$ from these leaf embeddings. Figure 1 illustrates the autoencoding process.

Intuitively, this means that the model starts from random embeddings, and therefore an essentially random merge order. Throughout training, tokens which are often part of the same merges will have their representations drawn together, so the representation reflects what they are likely to compose with. The model can then leverage any regularities to better perform reconstruction. This leads the representations to further reflect likely compositions and consequently increases the regularity in the structure. Ultimately, this leads to representations which must, by virtue of the training procedure, reflect the compositional semantics learned by the model.

For a more formal description of the operation of the model we begin by noting that the model generates *two* sets of embeddings. One set going up from leaves to root, and another coming back down from root to leaves. We denote these $\bar{e}$ and $\underline{e}$ respectively We also note that an embedding is typically viewed as $e \in \mathbb{R}^{U \times K}$ with K independent channels—of particular relevance to the composition and decomposition functions which act independently over the channels. Tokens are denoted as the vertices $w_i \in \Delta^V$ in a V-simplex for vocabulary size V. All together, the functioning of the model is then characterised by:

$$\Omega_\Psi(w_i) = w_i \, \Psi, \quad \Psi \in \mathbb{R}^{V \times (U \star K)} \tag{1}$$

$$C_\Phi(\bar{e}_i, \bar{e}_{i+1}) = \text{HCAT}(\bar{e}_i, \bar{e}_{i+1}) \, \Phi + \phi \quad \Phi \in \mathbb{R}^{2U \times U}, \phi \in \mathbb{R}^{U} \tag{2}$$

$$D_\Theta(\underline{e}_i) = \text{HSPLIT}(\underline{e}_i \, \Theta + \theta) \quad \Theta \in \mathbb{R}^{U \times 2U}, \theta \in \mathbb{R}^{2U} \tag{3}$$

$$\Lambda_\Gamma(\underline{e}_i) = \underline{e}_i \, \Gamma \quad \Gamma \in \mathbb{R}^{(U \star K) \times V} \tag{4}$$

Given the nature of the model, a straightforward objective would be to simply reconstruct the tokens, formulated for sentence $\mathbf{w}$ and prediction $\widehat{\mathbf{w}}$ as $\mathcal{L}_{\text{CE}}(\mathbf{w}, \widehat{\mathbf{w}}) = -\frac{1}{N} \sum_{n=1}^{N} w_n \cdot \log \widehat{w}_n$. An alternate approach developed by Opper et al. (2023) leverages the multi-level structure of the model to define a contrastive objective over a batch of sentences $\{\mathbf{w}_b\}_{b=1}^{B}$ with a total of M nodes (internal + leaves). Noting that the up and down trees share the same underlying structure (modulo reversed edges), this objective draws together corresponding up and down embeddings at a given tree position, whilst pushing away other embeddings across the batch, using the cosine similarity

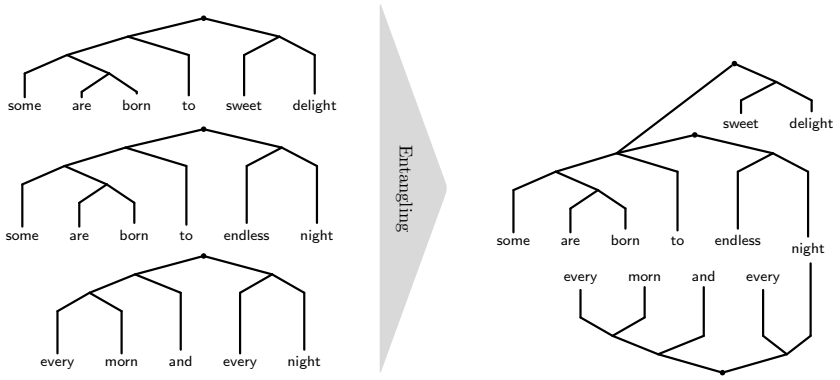


Figure 2: Entangled trees: Example of disjoint trees being transformed into an entangled tree. We leave out denoting functions from Eqs. (1) to (4) to avoid clutter and assume them implicitly present.

metric. Denoting the pairwise similarity matrix $A \in \mathbb{R}^{M \times M}$ between upward embeddings $\langle \bar{e}_i \rangle_{i=1}^{M}$ and downward embeddings $\langle \underline{e}_i \rangle_{i=1}^{M}$, and $A_{i\bullet}, A_{\bullet j}, A_{ij}$ the i^{th} row, j^{th} column, and $(i,j)^{\text{th}}$ entry of the matrix respectively, the objective is defined as: $\mathcal{L}_{\text{CO}}(\bar{e}, \underline{e}) = \frac{-1}{2M} \sum_{i=1}^{M} \log\left(\sigma_\tau(A_{i\bullet})\, \sigma_\tau(A_{\bullet i})\right)$ with tempered `softmax` $\sigma_\tau(\cdot)$ (temperature τ) normalising over the unspecified ($\bullet$) dimension.

4 Model

A particularly interesting characteristic of learning with explicitly structured models such as `Self-StrAE` or even earlier models such as the `IORNN` (Le & Zuidema, 2014) is the dichotomy between the upward and downward embeddings. Given their construction, the upward embeddings are always *locally-contextual*: they only encapsulate the context of the span they cover. For example, following Fig. 1, the upward embedding $\bar{e}$ for the span ‹ate doughnuts› is always the same regardless of context, no matter who did the eating. In contrast, downward embeddings are always *globally-contextual*: they must encapsulate the surrounding context by virtue of being decomposed from larger spans. For our example, this implies that there are multiple downward embeddings $\underline{e}^y$ for the given span, one for each $y \in \{\text{Lisa, Homer, } \dots\}$. To learn effective embeddings then, one must *marginalise* over these different downward embeddings to ensure that their meaning resolves over all these contexts.

4.1 From trees to entangled trees

We want to have the composition embeddings amortise over all possible contexts, and simultaneously we want all decompositions embeddings to resolve to the same thing. The representation of an entity Lisa should encapuslate everything she could possibly eat. Simulteanously if we take the average of everything she could eat we should get back to Lisa. Self-StrAE does not explicitly model this behaviour in its structure. Decomposition embeddings of the same entity only interact when we calculate the loss. On top of this, because the loss is taken over the batch, they are actually treated as false negatives to each other. Even though they are terms that ought not be pushed away, the objective ask them to be.

Algorithm 1 Banyan: Entangled Compose

Input: Global frontier $\langle (s_n, e_n) \rangle_{n=1}^{N}$, compose ($\circ$), concat ($\diamond$), similarity $\textsc{cSim}(e, e')$

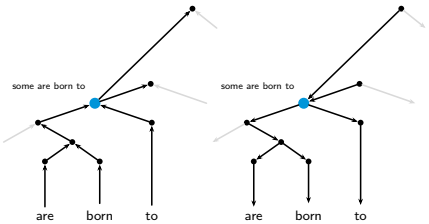
$\mathcal{A} \leftarrow \langle (s_n, e_n) \rangle_{n=1}^{N}$ ▷ initialise frontier
$(\mathcal{V}, \mathcal{E}) \leftarrow (\varnothing, \varnothing)$ ▷ initialise graph
while $\exists i : s_i \diamond s_{i+1} \notin_s \mathcal{V}$ **do**
  $i^\star \leftarrow \arg\max_i \textsc{cSim}(e_i, e_{i+1})$
    ▷ location of closest adjacent pair
  $e_p = \circ(e_{i^\star}, e_{i^\star+1})$ ▷ compute composition
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{(s_{i^\star} \diamond s_{i^\star+1}, e_p)\}$
  $\mathcal{E} \leftarrow \mathcal{E} \cup \{p \sim i^\star, p \sim (i^\star + 1)\}$
  $\mathcal{J} \leftarrow \{j : (s_j, s_{j+1}) = (s_{i^\star}, s_{i^\star+1})\}$
    ▷ locations of all occurrences of this pair
  $\mathcal{A} \leftarrow \mathcal{A} \setminus \{\forall_{j \in \mathcal{J}}\, \mathcal{A}_j, \mathcal{A}_{j+1}\}$
    ▷ delete occurrences from those locations
  $\mathcal{A} \leftarrow \mathcal{A} \cup_{\mathcal{J}} \{(s_{i^\star} \diamond s_{i^\star+1}, e_p)\}$
    ▷ insert composition into those locations
return Graph $(\mathcal{V}, \mathcal{E})$

Our innovation here is to address both these issues together by formulating the process in terms of entangled trees—where entangling refers to reduction of a set of disjoint tree structures into a single conjoined graph structure. An example is shown in Fig. 2 with disjoint trees on the left and resulting entangled tree on the right. Here, all instances of ‹night› and ‹some are born to› are captured by a single node

representing that constituent. We call our model Banyan on account of this entangling, because, like the tree, it can have many roots -consisting of nodes frequently reused across contexts.

Entangling: Constructing an entangled tree given a set of disjoint trees is a relatively straightforward process and is formally specified in Algorithm 1. In contrast to the agglomerative clustering employed in Self-StrAE, here we employ a global frontier spanning all leaf nodes across the given data. The key differences to the prior methods are mainly to do with constructing a graph jointly with progressing the frontier and ensuring that new nodes are never duplicated, for which we employ a node identity s_n in addition to the node embedding e_n.

Incorporating context Following the entangling of trees described, the model proceeds in a similar vein to Self-StrAE, by composing upwards from leaves to roots (multiple roots corresponding to multiple trees), and then decomposing downwards back to the leaves. With entangled trees, while traversing upwards each node is always composed from the same two children, but on the way back down, things are different as each separate context for a given node provides a different downward embedding. This is shown in Fig. 3 focussing on a subgraph of the entangled tree from Fig. 2(right). Note that the node in question (in blue) corresponds to the span ‹some are born to›, and has downward embeddings that incorporate context both from ‹endless night› and ‹sweet delight›. This is exactly as desired, as Banyan allows explicit aggregation to derive the downward embedding that resolves over the contexts. For any upward embedding $\bar{e}$ whose span occurs in different contexts $y \in \mathcal{Y}$, the corresponding downward embedding is derived by simply averaging over the different contextual down embeddings; i.e., $\underline{e} = 1/|\mathcal{Y}| \sum_y \underline{e}^y$.



(a) upward composition (b) downward decomposition

Figure 3: Upward and downward traversals for a section of the entangled tree from Fig. 2.

Effectiveness and efficiency Beyond the ability to explicitly incorporate context across data, entangled trees also help the contrastive objective by avoiding false negatives since they do not admit duplicate nodes by construction. Furthermore, the lack of duplicate nodes also drastically impacts the memory footprint of the model as one deals with the *set* of all nodes rather than counting each instance as its own node. These effects becomes more pronounced when entangling a larger set of instances as the likelihood of false negative and duplicates goes up together.

Practical estimation Given the advantages conferred by entangled trees, one would ideally want to construct it over *all* the available data. This however is not practically feasible as the size of data typically grows exponentially with time. To address this, we construct our model to estimate the given objective by taking steps over *batches* of data that are of a more manageable size, noting that this estimator is unbiased. To see this is the case, note that entangled trees only affects the downward embeddings directly, and that batching simply means that the resolved embedding is an average over *samples* instead of over all the data (*population*)—the sample mean is always an unbiased estimator of the population mean.

4.2 Simplified Message Passing

Complementary to the development of entangled trees to incorporate context, we also explore avenues to improve the message passing with the composition (C) and decomposition (D) functions. The original formulations of these from Eqs. (2) and (3) employ concatenation and splitting along with simple single-layer linear neural networks. The authors found that these simpler formulations led to better representations than e.g., Tree-LSTM cells, because they forced the model to conform to the compression order of the structure.

But if all we need for success is to respect the compression order, then we could possibly do better with an even simpler solution that exploits diagonalised functions (Ba et al., 2016)? These have become a hallmark of the recent resurgence in recurrent neural networks (Peng et al., 2023; Orvieto et al., 2023; De et al., 2024), by introducing decayed memory in the temporal dimension. Such a parameterisation means that rather than using full matrices as our C and D functions, we instead define them as:

$$C(\bar{e}_i, \bar{e}_{i+1}) = (\bar{e}_i \cdot \sigma(\Phi_l) + \bar{e}_{i+1} \cdot \sigma(\Phi_r)) + \phi \quad \Phi_l, \Phi_r, \phi \in \mathbb{R}^U \tag{5}$$

$$D(\underline{e}_i) = \left(\underline{e}_i \cdot \sigma(\Theta_l) + \theta_l, \;\; \underline{e}_i \cdot \sigma(\Theta_r) + \theta_r\right) \quad \Theta_l, \Theta_r, \theta_l, \theta_r \in \mathbb{R}^U \tag{6}$$

with sigmoid non-linearity (σ) applied to parameters both for numerical stability and to make the functions enforce a decayed memory over structure depth. The repeated application of the diagonal composition function will decay the influence of nodes further down in the tree, thereby respecting the compression order of the structure. In addition, during composition parent representations can increase in magnitude as they are the sum of the two children. During decomposition child representations will, by necessity, reduce back down in magnitude towards the core. In this way the functions further mimic the information flow specified by the entangled trees.

These relatively simple changes have a pretty drastic effect, both in terms of performance (see experiments) as well as memory footprint, with parameters now reduced by a factor of U compared to the functions from Eqs. (2) and (3)

5 EXPERIMENTS:

5.1 WARMUP: ENGLISH LANGUAGE EVALUATION

Goal: Having outlined Banyan, we want to test whether it can efficiently learn semantics. We start by evaluating on English, as there are far more test sets available than for low resource languages.

Evaluation: We want to evaluate how well Banyan is able to learn effective semantic representations. Ideally we want to probe this at different levels of hierarchy, covering both the lexical and sentential level. Our evaluation is unsupervised, both to directly probe the effect of pretraining with the inductive bias, and because this setting has greater parity to what may be expected in a low resource domain, where there are few labelled datasets. For these reasons, we turn to a series of tasks which measure correlation between cosine similarity of embedding pairs for two examples and human judgements of their semantic correspondence. On the word level, we use Simlex-999 (Hill et al., 2015) and WordSim-S/R (Agirre et al., 2009). All tasks measure semantics, but do so on differing axes. To understand this, we must first qualify the difference between semantic similarity and relatedness. Semantic similarity measures the extent to which entities act the same way. For example, 'running' and 'singing' are similar as they share the role verb. Semantic relatedness measures conceptual association. For example, 'singing' and 'fame' may be highly related. Simlex measures similarity at the exclusion of relatedness. Wordsim S measures similarity without penalising relatedness. And Wordsim R measures relatedness. On the sentence level, we use STS-12 through 16 (Agirre et al., 2012; 2013; 2014; 2015; 2016), the STS-B (Cer et al., 2017), SICK-R (Marelli et al., 2014) and SemRel (Ousidhoum et al., 2024). Each measures slightly different aspects of sentential semantics, covering similarity, relatedness, equality and entailment. A good model should do well on all of them.

Baselines: We compare against the `Self-StrAE`, `GloVe` embeddings (Pennington et al., 2014) and a `RoBERTa` (Liu et al., 2019) in the medium configuration from (Turc et al., 2019). `Self-StrAE` stands as the closest point of comparison to `Banyan`. `Self-StrAE` indicates the performance level of structured representation learning lies, as well as any improvements we are able to achieve. `GloVe` lets us compare to traditional static embeddings. This comparison probes whether our model is learning anything more than just simple bag of word features. To obtain sentence embeddings, we report results using both the simple average of the word embeddings and the average with filler words removed following (Reimers & Gurevych, 2019). These filler words contribute little semantic information and their removal has been shown to improve performance. For `RoBERTa`, we report results using both the standard model, and again after enhancing `RoBERTa` through an extra round of contrastive `SimCSE` training (Gao et al., 2021), as a further STS baseline. In both cases, we generate sentence embeddings through mean pooling. To produce static embeddings from `RoBERTa` to use in lexical evaluation, we follow Bommasani et al. (2020) and average the contextualised representations of all occurrences of the word in the training set. The `RoBERTa` is intended as a stronger baseline. It has significantly more parameters than `Banyan` and is able to model meaning in context unlike `GloVe`.

Hyperparameters and Pre-training Details: For all models we set the embedding size to 256. For `Self-StrAE` we use the configuration of (Opper et al., 2023) and set embeddings as square matrices (i.e., K=16 and U=16). For `Banyan` we set these values to K=128 and U=2, because the more

Table 1: Sentence level results for models pretrained on English. Higher is better. Results represent the average across four random initialisations. Only columns where there is no standard deviation overlap between models are bolded. Spearman's ρ is * 100 following convention.

Model	STS-12	STS-13	STS-14	STS-15	STS-16	STS-B	SICK-R	SemRel	Score
Self-StrAE	31.98 ± 0.58	53.88 ± 0.68	37.73 ± 0.70	55.23 ± 0.58	55.55 ± 0.47	39.53 ± 1.61	51.78 ± 0.29	50.05 ± 0.92	46.59 ± 0.43
GloVe	31.61 ± 0.31	21.69 ± 0.12	27.37 ± 0.10	40.42 ± 0.09	29.27 ± 0.12	28.25 ± 0.08	50.20 ± 0.25	41.20 ± 0.43	33.75 ± 0.04
+ stopword rm	39.00 ± 0.57	41.61 ± 0.19	39.31 ± 0.18	51.06 ± 0.35	45.14 ± 0.14	48.40 ± 0.07	52.80 ± 0.04	42.37 ± 0.13	44.96 ± 0.10
RoBERTa	42.77 ± 1.27	51.70 ± 1.30	45.67 ± 1.42	63.97 ± 0.81	59.60 ± 0.61	39.97 ± 0.95	52.93 ± 0.23	52.73 ± 0.58	51.08 ± 0.61
+ SimCSE	50.63 ± 1.45	62.23 ± 2.51	54.17 ± 2.10	68.77 ± 3.00	66.67 ± 1.40	53.53 ± 1.18	**56.87 ± 1.16**	59.27 ± 0.93	59.02 ± 1.45
Banyan	51.20 ± 0.007	**69.10 ± 0.002**	**63.20 ± 0.004**	**73.20 ± 0.002**	66.60 ± 0.002	**61.50 ± 0.002**	55.50 ± 0.003	**61.60 ± 0.002**	**62.70 ± 0.001**

independent channels we allowed the better the model seemed to perform. We refer the reader to the Appendix for ablations. We also note that because we can perform this reduction in channel size, the number of non-embedding parameters for Banyan drops to just 14, as these are directly proportional to U. We trained Self-StrAE and Banyan for 15 epochs (circa 15k steps and sufficient for convergence) using the Adam optimizer (Kingma & Ba, 2015), with a learning rate of 1e-3 for Banyan and 1e-4 for Self-StrAE using a batch size of 512. We applied dropout of 0.2 on the embeddings and 0.1 on the composition and decomposition function outputs. The temperature hyper-parameter for the Self-StrAE was set to 0.2. To process the graphs we used DGL (Wang et al., 2020). The GloVe baseline was trained for 15 epochs with a learning rate of 1e-3, and a window size of 10. We used the official C++ implementation. RoBERTa medium was trained for 200,000 steps, (10% of which were used for warmup). We used a learning rate of 5e-5, and a linear schedule. Positional embeddings are relative key-query. The configuration for RoBERTa medium is 8 layers, 8 attention heads and 2048 dimensional feedforward layers. We used the Transformers library to implement and train the model (Wolf et al., 2020). For SimCSE training, we used the default parameters and the official implementation for unsupervised RoBERTa training from Gao et al. (2021). As our pre-training corpus we selected the WikiText-103 benchmark dataset (Merity et al., 2016). The RoBERTa and GloVe baselines are trained on the full corpus (103 million tokens), representing the upper-middle end of the level of data scale that might be available for a language. Whereas we trained Self-StrAE and Banyan on a uniform subsample of 10 million tokens, representing the lower end of how many tokens might be available, because these explicit structure models are supposed to be efficient learners.

Results: Results are shown in Tables 1 and 2. On both the word level and sentence level Banyan does much better than Self-StrAE. We ablate the reasons for this in more detail later in the manuscript. Both models suffer on SimLex because they need to model both similarity and relatedness as the latter dictates merge (related concepts often compose together). However, the important thing to note is that the structured models effectively transfer the same performance from the word level to the sentence level. They can take advantage of composition, and transfer the meaning of the parts to understanding the meaning of the whole. The GloVe baseline is good on the word level, but does not generalise to the sentence level as well as the transformer, even when we give it stopword removal. It cannot transfer semantic knowledge seamlessly to different levels of complexity. Banyan can, and is able to achieve comparable or better performance than the SimCSE RoBERTa despite being much smaller and exposed to 10x less pre-training data. This means we have a structured model that remains efficient and cheap, and also effective at representation learning.

Table 2: Word level results analogous to Table 1.

Model	Simlex	Wordsim-S	Wordsim-R	Score
Self-StrAE	13.80 ± 0.41	54.38 ± 0.78	52.85 ± 1.27	40.34 ± 0.66
GloVe	27.47 ± 0.25	62.53 ± 0.42	51.00 ± 0.56	47.00 ± 0.38
RoBERTa	**29.23 ± 0.64**	61.97 ± 2.38	46.00 ± 2.13	45.73 ± 1.71
Banyan	16.57 ± 0.02	**63.25 ± 0.03**	**69.00 ± 0.01**	**49.61 ± 0.02**

5.2 Multilingual Evaluation:

Goal: From the results in English we know that Banyan is an efficient learner: it can produce good representations without requiring large-scale data or compute. This implies potential use for under represented communities, whose languages are not well covered by current NLP approaches. Now we have to test that.

Evaluation: Learning semantic representations for low resource languages remains an ongoing challenge in NLP. A core problem is not just the lack of training data, but also the lack of evaluation datasets. Recent work by Ousidhoum et al. (2024) has sought to address this issue, providing semantic relatedness test sets for several low resource Asian and African languages. These test sets are evaluated the same as before, comparing the cosine similarity between model embeddings for

Table 3: Multilingual Results. Banyan performance is taken over four random seeds. Baselines marked with † have been finetuned on supervised semantic similarity datasets. FT denotes unsupervised finetuning using masked language modelling on the same corpora as Banyan.

Model	Indonesian	Arabic	Telugu	Marathi	Mor. Arabic	Kinyarwanda	Hausa	Afrikaans	Spanish	Amharic	Hindi	Score
XLM-R	46.7	31.6	46.3	55.7	17.4	13.2	4.1	56.2	68.9	57.3	52.7	40.92
Llama-3.1 (8B)	**53.4**	31.1	65.6	63.4	19.4	19.7	6.1	65.4	66.7	64.1	61.7	46.96
Mistral Nemo	50.7	20.1	57	52.3	15.1	16.3	1.8	58.3	66.2	53.2	55.8	40.62
MiniLM-L12†	39	16.1	34.8	39.5	13.5	35	32.7	74.1	58.8	9.6	43.8	36.08
Paraphrase XLM-R†	46.1	**61**	58.1	**79.6**	7.1	43.2	22.5	76.8	71.7	64.6	52	52.97
XLM-R (FT)	47.9	33.6	68.8	75.1	21.6	19.4	14.6	72.6	**72.8**	59.6	57.6	49.41
Banyan	44.17 ± 1.11	43.20 ± 1.82	**71.13 ± 0.91**	67.67 ± 0.64	**52.00 ± 2.25**	**46.1 ± 0.32**	**43.7 ± 1.21**	**78.68 ± 0.30**	60.95 ± 0.76	**66.18 ± 0.46**	61.83 ± 0.6	**57.78**

two sequences with human judgements of their semantic match. As before, evaluation is zero-shot unsupervised. Allowing us to evaluate Banyan on Indonesian, Arabic, Telugu, Marathi, Moroccan Arabic, Kinyarwanda, Hausa, Afrikaans, Spanish, Amharic and Hindi. These represent a spectrum in terms how well resourced they are. For example, Spanish and Hindi are reasonably well represented, while Moroccan Arabic and Kinyarwanda have extremely little training data.

Baselines: We select XLM-R (Conneau et al., 2019): a transformer encoder trained on 2TB of multilingual data. Llama 3.1 8B (Dubey et al., 2024): a decoder only LLM trained on 15 trillion tokens. Mistral Nemo 12B: a decoder only LLM designed with multi-lingual capacities in mind. In addition we also compare against two specialised embedding models from the sentence transformers range (Reimers & Gurevych, 2019): Mini-LM-L12-V2 and Paraphrase-XLM-R-Multilingual-V1. These are pre-trained transformer encoders that have been finetuned on supervised datasets designed to produce high quality semantic representations. The baselines we select here are emblematic of the kind of models one might reach for in order to embed a corpus. For all models we use mean pooling to produce the sentence representation following Reimers & Gurevych (2019); Li & Li (2024). Finally, for parity we include an XLM-R baseline which is finetuned on the same corpora.

Banyan Pre-training and Hyperparameters: For Afrikaans, Spanish and Amharic we obtained corpora from Leipzig Corpora Collection[1] (Goldhahn et al., 2012). For Amharic we utilised a MiT licenced pre-training set of 1 million sequences available on the Huggingface hub at this link. Kinyarwanda and Hausa data was sourced from Opus (Nygaard & Tiedemann, 2003). Each dataset consists of roughly 10 million tokens. We utilise a pre-trained BPE tokenizer for each language from the BPEMB Python package (Heinzerling & Strube, 2018). Though the package also provides pre-trained embeddings, we solely use the tokenizer and learn embeddings from scratch. For the model hyperparameters we keep all the settings from the experiments on English. For XLM-R we finetune for up to 100k steps with early stopping, using a linearly scheduled learning rate of 5e-5 with 10 percent of stepping serving as warmup. XLM-R runs at batch size 128 across 4xA40 45gb cards.

Results: See Table 3. In Spanish, a well resourced language with high coverage, the transformer baselines almost all outperform Banyan. However, as languages become lower resourced the picture changes, and Banyan outperforms or is comparable to the baselines. This even includes the multilingual XLM-R that has undergone supervised training to produce better representations. While finetuning XLM-R improves performance the amount of benefit it provides is not uniform and is insufficient to prove viable in the very low resource cases. Banyan is able to learn competitive representations consistently across languages, unsupervised and with very little data, meaning it provides a viable alternative for producing embeddings cheaply and efficiently for low resource languages.

5.3 Efficiency

Alongside its embedding matrix, Banyan has two central components: the composition and decomposition functions. We diagonalise these functions so that they are both easier to compute and have fewer parameters than standard weight matrices, ($2U$ rather than $2U \times U$), achieving a further order of magnitude reduction in parameters compared with the already minimal Self-StrAE.

Secondly, by exploiting entangled tree structure the number of nodes grows at a significantly reduced rate with batch size compared with standard sentential trees (see

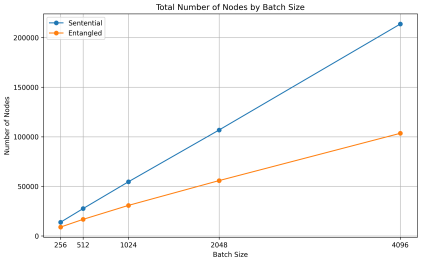


Figure 4: Total #nodes in entangled trees vs sentential trees as batch size grows.

[1]For Spanish and Hindi we select the mixed corpus and uniformly subsample to reduce size to ≈10M tokens.

Table 4: Ablations of modelling changes made for Banyan. Higher is better. Results represent the average across four random initialisations. Only columns where there is no standard deviation overlap between models are bolded. Spearman's ρ is * 100 following convention.

Model	STS-12	STS-13	STS-14	STS-15	STS-16	STS-B	SICK-R	SemRel	Score
Standard Trees	31.98 ± 0.58	53.88 ± 0.68	37.73 ± 0.70	55.23 ± 0.58	55.55 ± 0.47	39.53 ± 1.61	51.78 ± 0.29	50.05 ± 0.92	46.59 ± 0.43
+ diag functions	35.13 ± 0.33	56.05 ± 0.24	40.58 ± 0.05	58.83 ± 0.10	56.78 ± 0.21	44.10 ± 0.14	53.35 ± 0.17	52.65 ± 0.17	49.68 ± 0.06
++ CE loss	47.10 ± 1.04	61.85 ± 1.44	58.60 ± 1.34	70.45 ± 0.57	62.45 ± 0.70	59.50 ± 0.53	$\mathbf{59.00 \pm 0.26}$	60.33 ± 0.26	59.91 ± 0.54
Entangled Trees	38.98 ± 0.39	61.75 ± 0.14	43.65 ± 0.46	58.21 ± 0.41	55.29 ± 0.23	46.15 ± 0.71	53.93 ± 0.16	52.53 ± 0.09	51.31 ± 0.13
+ diag functions	44.15 ± 0.002	62.80 ± 0.002	48.30 ± 0.001	64.60 ± 0.002	60.30 ± 0.001	49.80 ± 0.002	55.14 ± 0.001	57.70 ± 0.001	55.23 ± 0.001
++ CE loss	$\mathbf{51.20 \pm 0.007}$	$\mathbf{69.10 \pm 0.002}$	$\mathbf{63.20 \pm 0.004}$	$\mathbf{73.20 \pm 0.002}$	$\mathbf{66.60 \pm 0.002}$	$\mathbf{61.50 \pm 0.002}$	55.50 ± 0.003	$\mathbf{61.60 \pm 0.002}$	$\mathbf{62.70 \pm 0.001}$

Fig. 4). This is because the number of reused constituent nodes also grows as batch size increases, and entangled trees capture the set of all constituents, which consequently does not grow as drastically. In practical terms, because entangled trees requires fewer nodes, and each node requires two distinct embeddings ($\bar{e}$ and $\underline{e}$) to be held for it, reducing the number of nodes required leads to radical improvements in memory efficiency. Put together, these changes mean that we can train `Banyan` very quickly as we can use large batches and its small number of parameters ensure quick convergence. On a single Nvidia A40 GPU with a batch size of 1024, Banyan pretrains from scratch in under 50 minutes, meaning that the total cost of pretraining a `Banyan` model sits at around 30 cents[2]. Free-tier Google Colab users can achieve similar results in about two hours with a smaller batch size. Inference can also be performed on CPU on typical laptops, because the model is so small. Combined with its data efficiency, we believe this provides a promising alternative for low resource languages and communities.

5.4 Ablations

We have shown that `Banyan` is more effective than its `Self-StrAE` predecessor, but what is the impact of the different modelling changes we made? To test this we ablate our results from our first set of experiments on English (see Table 4).

The simplest positive impacts to see are from the introduction of the diagonalised composition and decomposition functions. These are sigmoided scalar values with which we multiply embeddings. Therefore they act similarly to the fast weights of Ba et al. (2016), decaying in the influence of embeddings further down in the tree on the root representation. This means that the embeddings produced by the model are restricted to conform to the compression order dictated by the structure, and we know from Opper et al. (2023), that the more we can enforce this constraint the better our representations will end up. Secondly, such simple message passing functions bias the representation space towards informative separability. There has to be some signal from which to perform reconstruction, and all the pressure is now on the representations.

Switching the objective to cross entropy over the vocabulary rather than the contrastive objective used by Opper et al. (2023) also yields significant benefits. This is likely because the contrastive loss is supposed to be beneficial because it enforces a pressure for representations to be uniformly distributed in space (Wang & Isola, 2020). However, our other modelling changes already push towards this quality. While it is a shame because the contrastive loss is conceptually elegant. It is known to have problems and eventually lead to shortcut solutions (Robinson et al., 2021). Therefore having a more robust objective grounded in data, like the cross entropy over the vocabulary, is actually quite nice.

Finally, changing to entangled trees is also beneficial. The effect is more pronounced before switching to the cross entropy objective, as it removes the issue of false negatives as discussed in Section 4. However, it also is beneficial beyond this. Entangling explicitly allows the model to take advantage of shared constituency structure between complex sequences, because it combines the information from all incoming parent messages. The fact that performance improves using the cross entropy objective shows that explicitly allowing the model to take advantage of such systematicity is useful.

Table 5: Number of non-embedding parameters for models studied.

Model	`Banyan`	`Self-StrAE`	`RoBERTa (M)`	All-MiniLM-L12-V2	`XLM-R`	Llama 3.1	Mistral Nemo
Params	14	1072	≈10M	≈21M	≈85M	≈8B	≈12B

[2]Current cloud computing costs sourced from: https://www.runpod.io/pricing

6 Conclusion, Limitations and Future Work

We introduce Banyan, a Self-Structuring AutoEncoder. Banyan's focus on global, entangled structure and simplified message passing exploits the benefits of structured compositions inherent in language data. It is more effective and efficient than prior work from which we draw three central conclusions.

Firstly, explicitly modelling structured compositions is an effective inductive bias. Table 5 shows the parameters for the structured models versus the baselines. The structured models are far smaller, with tens or thousands of parameters instead of millions or billions. And nonetheless, Banyan is still competitive across several metrics, indicating we have found an efficient learning procedure.

Secondly, we have not yet fully exploited the potential of the inductive bias. Banyan still relies on greedy agglomerative clustering to induce structure. This is effective, but sub-optimal. Future work could make the structure induction procedure parametric and learnable. The type of structure models are exposed to impacts the quality of learnt semantic representations (Opper et al., 2023). So if *how* we induce structure improves, the model should learn significantly better representations.

Thirdly, while this paper focuses on recursive neural networks for the purposes of efficiency and low resource applicability, the method could in principle be applied to representations from pre-trained transformer models. Firstly, the transformers attention essentially defines a soft (fully connected) graph between tokens, which could serve as a more flexible basis for constructing Banyan's discrete structures. Moreover, the entangled tree structure essentially serves as a map of the conceptual associations learned by a model, and could provide an interesting probe into the representation space of pre-trained LLMs.

Finally, good and cheap embedding models are useful for many applications. For example, the digital humanities need to organise corpora of ancient languages, making it easier for researchers to access texts they need. But these corpora are small, and these languages are unlikely to be present in pretraining corpora of larger models. Banyan provides an efficient solution for producing representations for both these use cases and low resource languages and under represented communities more generally. To conclude, Banyan addresses the problem of efficient learning in low-resource settings.

REFERENCES

Mohamed Abdalla, Krishnapriya Vishnubhotla, and Saif Mohammad. What makes sentences semantically related? a textual relatedness dataset and empirical study. In Andreas Vlachos and Isabelle Augenstein (eds.), *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 782–796, Dubrovnik, Croatia, May 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.eacl-main.55. URL https://aclanthology.org/2023.eacl-main.55.

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. A study on similarity and relatedness using distributional and WordNet-based approaches. In *North American Chapter of the Association for Computational Linguistics (NAACL)*, pp. 19–27, 2009.

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pp. 385–393, USA, 2012. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. *SEM 2013 shared task: Semantic textual similarity. In Mona Diab, Tim Baldwin, and Marco Baroni (eds.), *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pp. 32–43, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/S13-1004.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2014 task 10: Multilingual semantic textual similarity. In Preslav Nakov and Torsten Zesch (eds.), *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 81–91, Dublin, Ireland, August 2014. Association for Computational Linguistics. doi: 10.3115/v1/S14-2010. URL https://aclanthology.org/S14-2010.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. SemEval-2015 task 2: Semantic textual similarity, English, Spanish and pilot on interpretability. In Preslav Nakov, Torsten Zesch, Daniel Cer, and David Jurgens (eds.), *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pp. 252–263, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.18653/v1/S15-2045. URL https://aclanthology.org/S15-2045.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 497–511, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/S16-1081. URL https://aclanthology.org/S16-1081.

Jimmy Ba, Geoffrey Hinton, Volodymyr Mnih, Joel Z. Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past, 2016. URL https://arxiv.org/abs/1610.06258.

Rishi Bommasani, Kelly Davis, and Claire Cardie. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4758–4781, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.431. URL https://aclanthology.org/2020.acl-main.431.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Workshop on Semantic Evaluation (SemEval)*, pp. 1–14, 2017.

N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124, 1956. doi: 10.1109/TIT.1956.1056813.

Jishnu Ray Chowdhury and Cornelia Caragea. Modeling hierarchical structures with continuous recursive neural networks. *CoRR*, abs/2106.06038, 2021. URL https://arxiv.org/abs/2106.06038.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019. URL http://arxiv.org/abs/1911.02116.

Stephen Crain and Mineharu Nakayama. Structure dependence in grammar formation. *Language*, 63 (3):522–543, 1987.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. Griffin: Mixing gated linear recurrences with local attention for efficient language models, 2024.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy, May 2006. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2006/pdf/440_pdf.pdf.

Andrew Drozdov, Patrick Verga, Mohit Yadav, Mohit Iyyer, and Andrew McCallum. Unsupervised latent tree induction with deep inside-outside recursive autoencoders. *CoRR*, abs/1904.02142, 2019. URL http://arxiv.org/abs/1904.02142.

Andrew Drozdov, Subendhu Rongali, Yi-Pei Chen, Tim O'Gorman, Mohit Iyyer, and Andrew McCallum. Unsupervised parsing with S-DIORA: Single tree encoding for deep inside-outside recursive autoencoders. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4832–4845, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.392. URL https://aclanthology.org/2020.emnlp-main.392.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit

Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang,

Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. SimCSE: Simple contrastive learning of sentence embeddings. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 6894–6910, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.552. URL https://aclanthology.org/2021.emnlp-main.552.

Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (eds.), *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pp. 759–765, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/327_Paper.pdf.

Zellig S. Harris. Distributional structure. *Word*, 10:146–162, 1954.

Serhii Havrylov, Germán Kruszewski, and Armand Joulin. Cooperative learning of disjoint syntax and semantics. In Jill Burstein, Christy Doran, and Thamar Solorio (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1118–1128, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1115. URL https://aclanthology.org/N19-1115.

Benjamin Heinzerling and Michael Strube. BPEmb: Tokenization-free Pre-trained Subword Embeddings in 275 Languages. In Nicoletta Calzolari (Conference chair), Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga (eds.), *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 7-12, 2018 2018. European Language Resources Association (ELRA). ISBN 979-10-95546-00-9.

Felix Hill, Roi Reichart, and Anna Korhonen. SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695, 2015.

Xiang Hu, Haitao Mi, Zujie Wen, Yafang Wang, Yi Su, Jing Zheng, and Gerard de Melo. R2D2: Recursive transformer based on differentiable tree for interpretable hierarchical language modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4897–4908, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.379. URL https://aclanthology.org/2021.acl-long.379.

Xiang Hu, Haitao Mi, Liang Li, and Gerard de Melo. Fast-R2D2: A pretrained recursive neural network based on pruned CKY for grammar induction and text representation. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 2809–2821, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.181. URL https://aclanthology.org/2022.emnlp-main.181.

Takuya Ito, Tim Klinger, Douglas H. Schultz, John D. Murray, Michael W. Cole, and Mattia Rigotti. Compositional generalization through abstract representations in human and artificial neural networks, 2022. URL https://arxiv.org/abs/2209.07431.

Yangfeng Ji and Jacob Eisenstein. One vector is not enough: Entity-augmented distributed semantics for discourse relations. *Transactions of the Association for Computational Linguistics*, 3:329–344, 2015. doi: 10.1162/tacl_a_00142. URL https://aclanthology.org/Q15-1024.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diega, CA, USA, 2015.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *CoRR*, abs/1604.00289, 2016. URL http://arxiv.org/abs/1604.00289.

Phong Le and Willem Zuidema. Inside-outside semantics: A framework for neural models of semantic composition. In *NIPS 2014 Workshop on Deep Learning and Representation Learning*, 2014.

Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *Association for Computational Linguistics (ACL)(Volume 2: Short Papers)*, pp. 302–308, 2014.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. *CoRR*, abs/2005.11401, 2020. URL https://arxiv.org/abs/2005.11401.

Xianming Li and Jing Li. Angle-optimized text embeddings, 2024. URL https://arxiv.org/abs/2309.12871.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pp. 216–223, Reykjavik, Iceland, May 2014. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2014/pdf/363_Paper.pdf.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *CoRR*, abs/1609.07843, 2016. URL http://arxiv.org/abs/1609.07843.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *International Conference on Learning Representations (ICLR)*, 2013.

Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D. Manning. Pushdown layers: Encoding recursive structure in transformer language models, 2023. URL https://arxiv.org/abs/2310.19089.

Lars Nygaard and Jörg Tiedemann. Opus—an open source parallel corpus. In *Proceedings of the 13th Nordic Conference on Computational Linguistics*, 2003.

Mattia Opper and Siddharth Narayanaswamy. Self-strae at semeval-2024 task 1: Making self-structuring autoencoders learn more with less. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pp. 108–115. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.semeval-1.18. URL http://dx.doi.org/10.18653/v1/2024.semeval-1.18.

Mattia Opper, Victor Prokhorov, and Siddharth N. Strae: Autoencoding for pre-trained embeddings using explicit structure. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7544–7560. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.469. URL http://dx.doi.org/10.18653/v1/2023.emnlp-main.469.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences, 2023.

Nedjma Ousidhoum, Shamsuddeen Hassan Muhammad, Mohamed Abdalla, Idris Abdulmumin, Ibrahim Said Ahmad, Sanchit Ahuja, Alham Fikri Aji, Vladimir Araujo, Meriem Beloucif, Christine De Kock, Oumaima Hourrane, Manish Shrivastava, Thamar Solorio, Nirmal Surange, Krishnapriya Vishnubhotla, Seid Muhie Yimam, and Saif M. Mohammad. SemEval-2024 task 1: Semantic textual relatedness for african and asian languages. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*. Association for Computational Linguistics, 2024.

Christophe Pallier, Anne-Dominique Devauchelle, and Stanislas Dehaene. Cortical representation of the constituent structure of sentences. *Proceedings of the National Academy of Sciences*, 108(6): 2522–2527, 2011. doi: 10.1073/pnas.1018711108.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Xiangru Tang, Bolun Wang, Johan S. Wind, Stansilaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui-Jie Zhu. Rwkv: Reinventing rnns for the transformer era, 2023.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Jishnu Ray Chowdhury and Cornelia Caragea. Beam tree recursive cells. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 28768–28791. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/ray-chowdhury23a.html.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

Joshua Robinson, Li Sun, Ke Yu, Kayhan Batmanghelich, Stefanie Jegelka, and Suvrit Sra. Can contrastive learning avoid shortcut solutions? *CoRR*, abs/2106.11230, 2021. URL https://arxiv.org/abs/2106.11230.

Laurent Sartran, Samuel Barrett, Adhiguna Kuncoro, Miloš Stanojević, Phil Blunsom, and Chris Dyer. Transformer grammars: Augmenting transformer language models with syntactic inductive biases at scale. *Transactions of the Association for Computational Linguistics*, 10:1423–1439, 2022. doi: 10.1162/tacl_a_00526. URL https://aclanthology.org/2022.tacl-1.81.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 151–161, 2011.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1631–1642, 2013.

Paul Soulos, Henry Conklin, Mattia Opper, Paul Smolensky, Jianfeng Gao, and Roland Fernandez. Compositional generalization across distributional shifts with sparse tree operations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=fOQunr2E0T.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Association for Computational Linguistics (ACL)*, pp. 1556–1566, 2015.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: The impact of student initialization on knowledge distillation. *CoRR*, abs/1908.08962, 2019. URL http://arxiv.org/abs/1908.08962.

Shikhar Vashishth, Manik Bhandari, Prateek Yadav, Piyush Rai, Chiranjib Bhattacharyya, and Partha Talukdar. Incorporating syntactic and semantic information in word embeddings using graph convolutional networks. In *Association for Computational Linguistics (ACL)*, pp. 3308–3318, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5998–6008, 2017.

Bin Wang, C.-C. Jay Kuo, and Haizhou Li. Just rank: Rethinking evaluation with word and sentence similarities. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6060–6077, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.419. URL https://aclanthology.org/2022.acl-long.419.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. Deep graph library: A graph-centric, highly-performant package for graph neural networks, 2020.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *CoRR*, abs/2005.10242, 2020. URL https://arxiv.org/abs/2005.10242.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles, 2023. URL https://arxiv.org/abs/2307.05596.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

A Appendix

A.1 The k and u balance

The change to diagonal composition functions allows us to reduce the number of total parameters while maintaining performance. This is because the number of parameters is directly proportional to channel size u. We show ablations for this finding in Table 6. Our findings are similar to those of Opper & Narayanaswamy (2024) the smaller the channel size the better the model performs, although in our case we keep things stable between seeds whereas for them when they simplified they faced issues with extreme instability during training. This is thanks to the new message passing functions.

Table 6: Performance Depending on k and u values using new functions. Scores are the average of four random seeds.

k	u	Lex Score	STS Score
4	64	42.9 ± 0.01	43.5 ± 0.04
8	32	43.2 ± 0.02	48.6 ± 0.01
16	16	47.02 ± 0.03	62.2 ± 0.01
32	8	49.2 ± 0.01	62.9 ± 0.01
64	4	48.7 ± 0.01	62.9 ± 0.01
128	2	49.61 ± 0.02	62.7 ± 0.001
256	1	48.7 ± 0.01	62.9 ± 0.001