Transfer of Structural Knowledge from Synthetic Languages

Mikhail Budnikov
School of Computer Science & Engineering
Constructor University
Bremen, Germany 28359
mbudnikov@constructor.university

Ivan Yamshchikov
CAIRO
THWS
Würzburg, Germany 97082
ivan.yamshchikov@thws.de

Abstract

This work explores transfer learning from several synthetic languages to English. We investigate the structure of the embeddings in the fine-tuned models, the information they contain, and the capabilities of the fine-tuned models on simple linguistic tasks. We also introduce a new synthetic language that leads to better transfer to English than the languages used in previous research. Finally, we introduce Tiny-Cloze Benchmark — a new synthetic benchmark for natural language understanding that is more informative for less powerful models. We use Tiny-Cloze Benchmark to evaluate fine-tuned models in several domains demonstrating that fine-tuning on a new synthetic language allows for better performance on a variety of tasks.

1 Introduction

Large language models (LLMs) are becoming increasingly powerful and useful. However, the role of data properties in model training and what exactly models learn from the training data remains to a large extent out of the scope of most LLM papers. Yet surprisingly pre-training a model on a simple algorithmic task can lead to improvements in natural language modelling (Min et al., 2023). Such insights can be used to improve the construction of data sets for language models. Therefore, exploring the mechanisms of knowledge transfer is an important open question.

Scaling language models is a popular way to improve their performance[1]. However, as the detailed analysis in Villalobos et al. (2022) shows, the amount of data, especially high-quality text data, is limited and will become the main bottleneck in the coming years.

Such circumstances motivate research into more data-efficient learning algorithms and a better understanding of the mechanisms of generalization and transfer learning (Surkov and Yamshchikov, 2024). After all, humans are exposed to orders of magnitude less data than modern frontier models, yet demonstrate strong performance across many domains and outperform machines in some areas, even considering recent algorithmic advances.

Inspired by this, Huebner et al. (2021) demonstrate that training RoBERTa (Liu et al., 2019) on language acquisition data, together with some tweaks to model architecture and training, leads to 6000$\times$ gains in data efficiency. Similarly, Eldan and Li (2023) achieve significant model compression while retaining the ability to produce fluent and coherent English by using a generated dataset of stories for children, i.e. with small vocabulary and simple plots. And Gunasekar et al. (2023) find that filtering for or generating data with higher educational value is also very helpful.

Thus, there is a growing body of evidence that the choice of data matters a lot and simply scrapping the data from the web is suboptimal. However, there is a limited understanding of what properties of the data are important in different training stages. Papadimitriou and Jurafsky (2020) show that pre-training the LSTM (Hochreiter and Schmidhuber, 1997) on structured but not linguistic data, such as MIDI music, Java code, or even nested parentheses, reduces its perplexity when testing on Spanish. Sinha et al. (2021) find that removing all word order information from the pre-training phase does not significantly affect the final performance, given a fine-tuning phase with the correct word order. Krishna et al. (2021) sample the training data from a completely artificial language consisting of random n-grams and observe that pre-training objectives that require processing this information somehow, such as copying sentences in the right order, still improve the performance of the model on summarization tasks compared to a randomly initialized version.

However, research in this area currently tends to

[1]For a detailed review of the other ways to increase LLM generalization potential we address the reader to Budnikov et al. (2024)

Proceedings of the 1st Joint Workshop on Large Language Models and Structure Modeling (XLLM 2025), pages 242–251
August 1, 2025

focus on reporting surprising observations rather than explaining them. Papadimitriou and Jurafsky (2020) illustrate such observations using Figure 1.

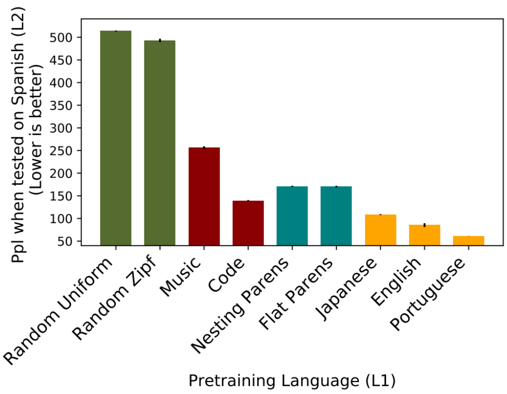

Figure 1: Perplexity on various types of input (Papadimitriou and Jurafsky, 2020).

This work attempts to build up on those observations and make a small further step studying the mechanisms of transfer learning.

In this paper, we address the following research questions:

- How do different synthetic pre-training datasets influence the complexity and transfer performance of language models on English tasks?
- What structural properties of the learned embeddings reflect the characteristics of the pre-training data?
- To what extent do synthetic pre-training languages affect the encoding of linguistic features in embeddings, as measured by linear probes?

Our contributions are summarized as follows:

- We introduce a new synthetic language, `flat_shuffle`, combining shuffle-based and bracket-based patterns, and compare it with existing synthetic datasets.
- We propose a transfer-learning-based measure to quantify language complexity and similarity via fine-tuning dynamics.
- We analyze the structure of embeddings through singular value spectra and clustering to assess their effective dimensionality.
- We employ linear probes to examine the linguistic information captured in embeddings fine-tuned on different synthetic languages.
- We present the Tiny-Cloze Benchmark, a synthetic NLU benchmark generated with GPT-4, demonstrating the benefits of synthetic pre-training.

First, as can be seen in the diagram above, different pre-training datasets, even if they all are unrelated to the target task, lead to different final performance. This suggests that some datasets are inherently more complex or more similar to the target language. We introduce a new synthetic "language" by combining ideas from the previous work and use it, as well as two existing synthetic datasets, to pre-train the models. We then fine-tune them on English using three different fine-tuning pipelines. We also provide an algorithm to assess the impact of the pretraining data on the resulting model parameters.

Second, since one of the settings for transfer learning involves fine-tuning only the embeddings, they are the natural target for investigation. We investigate the structure of the learned embeddings, namely the spectrum of their singular values to understand the effective dimensionality of the data, and the KMeans clustering of to check how uniformly the embeddings are distributed. To check what information is contained in the embeddings, we train linear probes to predict certain features of the words given their embeddings. Linear probes are a popular interpretability technique, but to our knowledge they have not been used to study the embeddings of models pre-trained on different datasets and fine-tuned to the same task.

Finally, we evaluate the performance of these models in natural language understanding. Since existing NLU datasets such as GLUE (Wang et al., 2018) and MMLU (Hendrycks et al., 2020) are designed for more powerful models, we use GPT-4 (OpenAI, 2023) to generate a similar benchmark consisting of 12 different subtasks[2].

2 Related work

One way of understanding the pre-training of language models is that we transfer some linguistic knowledge from a task with lots of available data

[2]To facilitate reproducibility and further research, we publish our code and data https://github.com/msh2481/language_transfer

to a downstream task (Han et al., 2021). The recent findings suggest that this is not the only relevant effect, and sometimes not even the most important one.

Papadimitriou and Jurafsky (2020), mentioned above, pre-trained an LSTM on structured but not linguistic data and found that adapting such a model to Spanish by fine-tuning only its input and output embeddings gave better perplexity than starting with a randomly initialised model. Their results and experimental setup established a framework that has been used in subsequent work, including this one. Ri and Tsuruoka (2022) improved these results replacing LSTM with a Transformer and changing the synthetic pre-training languages. Papadimitriou and Jurafsky (2023) used GPT-2 (Radford et al., 2019). Chiang and Lee (2022) introduce a family of Shuffle languages. Artetxe et al. (2019) used a similar technique to combine a task-specific corpus in English with a corpus in the target language unrelated to the task.

Such transfer also works in the opposite direction, from natural language to other domains. Lu et al. (2021) get performance comparable to training from scratch on different modalities by fine-tuning only the input and output embeddings of the pre-trained GPT-2. They also try different fine-tuning approaches, tuning the layer norm parameters and the last Transformer block in addition to the input and output embeddings.

Mehta et al. (2021) show that pre-training moves the model parameters into a flat basin of the loss landscape and suggest it as a reason why pre-trained models are less prone to catastrophic forgetting during fine-tuning. Neyshabur et al. (2020) also observe this and also show that models fine-tuned from the same checkpoint stay in the same basin. However, past data alone is almost never enough to predict unseen data, unless one makes some assumptions, i.e. "inductive bias". A useful inductive bias can be injected into the model by pre-training on data that has it. McCoy et al. (2020) use pre-training on natural languages with certain properties by model-agnostic meta-learning (Finn et al., 2017) to find which biases are needed to quickly acquire these languages. Wu et al. (2021) design synthetic datasets requiring deduction, induction, and abduction and pre-train on them to extract inductive bias for general mathematical reasoning. Lindemann et al. (2023) pre-train models to simulate finite state transducers given their description and achieve better generalization in NLP tasks with similar structure.

3 Synthetic Languages

In this paper we report a series of experiments with several synthetic languages. Following hyperparameter choices from Papadimitriou and Jurafsky (2023), for each of the languages described below, we use a sequence length of 512, a vocabulary size of 500, and generate $2 \cdot 10^6$ sequences so the total size of the dataset is approximately 10^9 tokens in each case.

We focus on three synthetic languages: `nested`, the k-Dyck nested bracket language; `flat`, the shuffle Dyck language with no nesting; and `flat_shuffle`, a block-wise shuffled variant of the flat language.

3.1 `nested`

Papadimitriou and Jurafsky (2020) used a stack-based grammar to generate sequences, where each token occurs twice and two pairs of tokens either do not intersect or one is nested in another. In other words, a balanced bracket sequence with multiple types of brackets.

Ri and Tsuruoka (2022) suggested using different tokens for opening and closing brackets, and found improved performance. We chose to implement this version, and save a synthetic language with 250 tokens for open brackets and 250 tokens for closing ones.

Tokens are generated sequentially, and on each step, a random decision is made whether to open a new bracket or to close an existing one. If the stack of open brackets is empty or there is not enough space before the end of the sequence, there is only one option. In other cases, an opening bracket is chosen with a probability of 0.4, and then the type of bracket is sampled uniformly.

Example word from `nested`:
<23 <42 <15 15> 42> 23> <56 56>

3.2 `flat`

This language is similar to the previous one. The only difference is that the nesting property can be violated.

In terms of sampling, it means that when a bracket should be closed, now there is more than one possibility. We select the bracket to close uniformly from all currently open ones.

Example word:

<23 <42 <15 42> 23> <56 15> 56>

3.3 `flat_shuffle`

The `flat_shuffle` language extends `flat` by partitioning bracket type IDs into contiguous blocks of size eight, such that each segment of 16 tokens is sampled exclusively from one block, yielding a permutation of those brackets within the segment. While the languages described above are each based on a single rule, this extension introduces additional structure to the data, which we hypothesize can improve model performance.

We suggest to use an idea of `shuffle` languages from Chiang and Lee (2022) as an extra pattern because it was orthogonal to the bracket balancing essence of the previous datasets. The combined dataset is based on `flat`, but each consecutive group of 16 tokens has a range of 8 bracket types assigned to it, and all brackets on this segment are sampled only from these types. That is, each such group is a permutation of the corresponding brackets.

It adds two interesting properties to the task of next token prediction. First, in the middle of the line the model has to look at previous tokens to guess the range of bracket types to predict the next token. Second, the model can guess increasingly more accurately by remembering which tokens were already used if we are close to the end of the permutation. In particular, the last token in each permutation can be predicted with certainty. Surprisingly, even small Transformer models were able to capture this pattern and indeed predicted the last token with close to zero loss.

Example word (purple and green parts represent two blocks, $[16, 20)$ and $[36, 40)$):

<16 <18 16> <17 <19 18> 17> <38 38> <36 19> <39 39> <37 36> 37>

4 Methodology

Some languages, both synthetic and natural, are more complex than others. For example, it is much easier to understand the concept of balanced bracket sequences than to learn Assyrian language. Moreover, some languages can be understood more easily if the learner already knows another language. For instance, humans need less effort to learn a language from the same language family, and large language models can be fine-tuned for a similar downstream task using much less data than was used for their pre-training.

One approach to formalize this intuition of complexity and similarity is the Chomsky hierarchy of languages (Chomsky, 1956). It formally defines several classes of grammars, each one strictly more general than the previous one, and the properties of these classes are very well understood. For example, `nested` is a context-free language, while `flat` is context-dependent. However, for languages from the same class, we need some other tool to find more fine-grained differences. We propose a transfer-learning based approach to quantify those.

An important observation is that transfer learning between languages is not symmetric, and it allows us to estimate both (relative) complexity and similarity of two languages. If languages are similar, transfer learning should go well in both directions. However, if one language is more complex than another, at least in the sense of having strictly more patterns, one would expect transfer learning to be much easier from the hard language to the easy one. So, assuming that we have some operationalization $f(A, B)$ of "difficulty of transfer learning from language A to language B", we can take $\frac{1}{2}(f(A, B) + f(B, A))$ to mean dis-similarity of A and B and $\frac{1}{2}(f(B, A) - f(A, B))$ to mean complexity of A relative to B.

Our proposed way to operationalize this notion of "difficulty" is to just use perplexity of the model pre-trained on language A and then fine-tuned to language B, with some of the weights frozen. By varying the subset of the weights allowed to be fine-tuned we can get a more complete picture, i.e. for some pair of languages just tuning the embeddings might already be enough, which would mean that they share most of the structure.

A more practically-oriented way to compare synthetic languages is to see which of them better prepare models to learning English. To test this we take models pre-trained on each of the synthetic languages, fine-tune them to English, and check their language understanding capabilities with cloze questions.

Finally, we study the structure of the embeddings in terms of effective dimensionality and number of clusters, and then explore what English word features are learned by models fine-tuned from each of the synthetic languages.

5 Experiments

For all experiments, we used the TinyStories-8M model (Eldan and Li, 2023).

5.1 Transfer Learning

We used three levels of trainable weight subsets:

E: Only input and output embeddings are tuned;
EL: **E** plus the affine parameters of LayerNorms;
ELT: **EL** plus the entire last Transformer layer;

For pre-training, we waited until convergence that was close to the theoretical lower bounds of loss or just long stagnation, which took 40K to 100K steps. The batch size was 8, and the sequence length was 512 tokens, so we used 160M to 400M tokens for pre-training. For fine-tuning, at each stage, we used a fixed amount of 12500 steps, the batch size was again 8, and the sequence length was 512 for bracket datasets and 128 for English (TinyStories), which means 51M and 13M tokens correspondingly. The learning rate was 10^{-3} for pre-training and $[10^{-2}, 2 \cdot 10^{-2}, 10^{-3}]$ for the three stages of fine-tuning.

Table 1 below presents the results of fine-tuning in both directions on certain pairs of languages.

The first row shows that `flat` is more complex than "nested". The second row demonstrates that `flat_shuffle` is more complex than `flat`. Indeed, fine-tuning in the direction `flat_shuffle` $\to$ `flat` $\to$ `nested` achieves relatively good performance already with the first stage of fine-tuning. The other experiments show that English is more complex than all synthetic languages used here, but it is also quite different, as the model needs more flexibility to adapt from English to `flat` or `flat_shuffle`.

5.2 Cloze Tests

To assess how well the models understand language in general, a different benchmark is needed. Since the models studied are too small for reliable question answering, reasoning, and other high-level cognitive skills, the test should be as simple as possible, ideally just measuring perplexity on some texts. There are existing datasets for natural language understanding, such as GLUE (Wang et al., 2018) and MMLU (Hendrycks et al., 2020), but they focus on more complex tasks.

Instead, we used GPT-4 (OpenAI, 2023) to generate Tiny Cloze Benchmark — a set of cloze[3] infilling questions in simple English. There are the following 12 subtasks, each with 10 cloze questions: 'synonyms and antonyms' — the model chooses one of two antonyms to correctly fill the gap in the sentence; 'Logical relations' — the model chooses a correct conjunction between two parts of the sentence; 'Subject-verb agreement' — the model chooses one of two verbs that corresponds to the given subject in the sentence; 'Prepositions' — the model chooses a correct preposition in the sentence; 'Conjunctions' — a task similar to 'Logical relations' but with different conjunctions; 'Temporal understanding' — filling in a correct temporal conjunction; 'Spatial understanding' — filling in a word based on spatial understanding of the sentence; 'Quantitative reasoning' — filling in the number into the sentence; 'Emotions' — filling the correct emotional adjective into the sentence; 'Narrative understanding' — filling one noun relevant for the narrative sentence; 'Ethics' — filling a noun for an ethical statement. You can find detailed examples of the tasks in Appendix.

Each cloze question consists of a prompt with a cloze marker, a correct answer, and an incorrect answer. For each question, the difference between log-probabilities of the correct and incorrect answers is measured and then averaged across each subtask. We measure the difference in log-likelihoods rather than accuracy, because it provides more information per sample, which is important given the relatively small size of our benchmark.

Here is an example question from the temporal understanding subtask:

```
[ "She ate breakfast # she went to
school", "before", "after", ]
```

For each of the synthetic languages, we used two models, one in which only the embeddings were fine-tuned on English (E), and another with all three stages of fine-tuning applied (ELT). We compared them with the model of the same architecture (8M parameters) trained on English from scratch and also to a four times larger model with 33M parameters trained on English from scratch to see which metrics can be improved.

As shown in Table 2, there are two interesting observations. First, models with all three stages of fine-tuning are better, predictably, than their counterparts having only the embeddings tuned, but this difference is more pronounced in `flat` and `flat_shuffle`. Second, Table 2 again shows the familiar pattern `nested <flat <flat_shuffle <scratch`, which proves the superiority of the introduced `flat_shuffle` dataset.

[3] https://en.wikipedia.org/wiki/Cloze_test

Language pair		L1 →				L2 →			
L1	L2	L2E	L2EL	L2ELT	L2Full	L1E	L1EL	L1ELT	L1Full
`nested`	`flat`	4.4	4.1	4.1	3.8	3.5	3.3	3.3	3.3
`flat`	`flat_shuffle`	2.5	2.4	2.2	2.0	3.8	3.8	3.7	3.8
`flat_shuffle`	English	2.4	2.3	2.0	1.2	2.8	2.6	2.1	2.0
`nested`	English	2.8	2.7	2.4	1.2	3.8	3.5	3.3	3.3
`flat`	English	2.7	2.6	2.4	1.2	4.3	4.2	3.8	3.8

Table 1: Pretraining on L1 and transferring to L2 and vice versa. Values are negative log-likelihoods in nats. "E", "L", and "T" indicate which layers were fine-tuned and stand for embeddings, LayerNorms, and (the last) Transformer block respectively. Columns ending in "Full" correspond to models trained from scratch on the respective language (i.e., no synthetic pretraining). We use the absolute difference of 0.2 nats per token as a threshold for "close performance".

	nested E	nested ELT	flat E	flat ELT	flat shuffle E	flat shuffle ELT	scratch 8M	scratch 33M
synonyms and antonyms	0.24	0.18	0.22	0.13	0.31	0.25	0.25	0.28
single - plural	0.08	0.15	0.19	0.50	0.03	0.33	0.58	0.71
logical relations	-0.08	-0.30	-0.44	-0.18	-0.13	-0.08	-0.04	0.09
subject-verb agreement	0.54	0.45	0.46	0.36	0.14	0.26	0.83	0.98
prepositions	0.43	0.52	0.51	0.53	0.40	0.48	0.94	1.12
conjunctions	0.46	0.43	0.45	0.38	0.36	0.49	0.63	0.82
temporal understanding	-0.13	-0.02	-0.14	0.04	0.09	0.36	0.44	0.73
spatial understanding	0.13	0.30	0.40	0.48	0.06	0.37	0.64	0.71
quantitative reasoning	-0.06	0.00	-0.14	-0.01	-0.14	-0.04	-0.04	-0.06
emotions	-0.08	0.03	0.05	0.07	0.20	-0.01	0.61	0.77
narrative understanding	-0.07	-0.04	0.03	0.07	0.04	0.04	0.17	0.27
ethics	0.32	0.17	0.34	0.22	0.27	0.30	0.25	0.51
Average	0.15	0.16	0.16	0.22	0.14	0.23	0.44	0.58

Table 2: Results on the Tiny-Cloze benchmark. Values show differences in log-likelihoods (in nats) between correct and incorrect answers. Fine-tuning on `flat_shuffle` gives the highest average score across three synthetic languages.

5.3 Dimensionality and Clusters

We hypothesize that models pretrained on more complex languages will exhibit a slower decay in the singular value spectrum, reflecting higher effective dimensionality required to encode richer structure.

The embedding dimension of the model used is $d = 256$, and human intuition, as well as many visualization techniques, work poorly for 256-dimensional vectors. We hypothesize that the singular value spectrum reflects the encoding complexity of the pre-training language: languages requiring richer structure will exhibit a slower decay in the spectrum tail, indicating higher effective dimensionality in the embeddings. Therefore, we employ two quantitative approaches.

First, for an $n \times d$ matrix of embeddings E, we consider its singular values (after zeroing out the mean of each column), or equivalently, the spectrum of the covariation matrix $A = E^T E$. The motivation behind this is that if all embeddings were contained in a k-dimensional subspace, and E had a rank k, then only k of the singular values would be nonzero. For real data, it is not the case, as all singular values are nonzero, but still, some directions have much larger variance than others, and the model is more likely to use features corresponding to those dimensions.

As we see in Figure 2, in models pre-trained on synthetic datasets, the spectrum is dominated by the first few dimensions. In particular, before fine-tuning, most of the interesting information about brackets is described by two axes: open-close and low-to-high bracket type id. While they learn more diverse features during fine-tuning on English, as described in the next sections, they still don't use the embedding space very efficiently. An interesting observation is how the tail of the spectrum behaves for models trained on different datasets: the spectrum of `flat` decays to zero slower than the one of `nested`, but the shape is similar, while the spectrum of `flat_shuffle` crosses `flat` at some point and behaves more similarly to the spectrum of the model trained on English from scratch.

Another interesting property is how the embeddings are clustered. To quantify it, we run k-means clustering for the embeddings varying the number of clusters and compare the plots of unexplained variance (Figure 3). Again, after pre-training on a synthetic language, the models have only two clusters: open and close brackets, and even after fine-tuning, the first few splits explain the majority of variance. Looking at the tail behavior, we observe a similar pattern: English is followed by `flat_shuffle`, then by `flat` and `nested`.

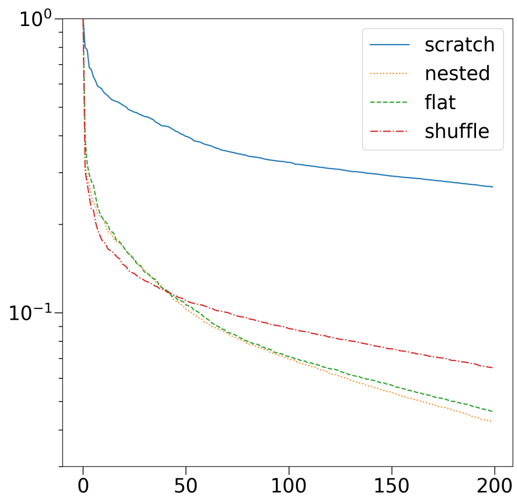

Figure 2: Spectrum of bracket embeddings

The `scratch` provides a reference for the embeddings on English. We can clearly see that `flat_shuffle` embeddings are characteristically different from `flat` and `nested` embeddings both in terms of the spectrum and in terms of the clusters

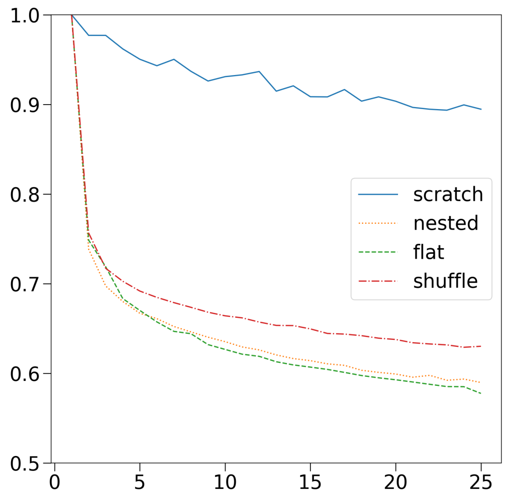

Figure 3: Clustering of bracket embeddings

they form.

5.4 Linear Probes for Word Features

We train all probes on embeddings obtained from embedding-only (E) fine-tuned models to isolate the impact of embedding space structure.

Now that we know something about the structure of the embedding space, a natural question to ask is how this structure is used. In other words, what information about a word can one extract from the embedding of the corresponding token?

Preliminary experiments showed that clusters of features correspond to properties like "noun", "3rd person verb", "adjective or adverb", etc. We hypothesize that embeddings from more complex synthetic pretraining (e.g., `flat_shuffle`) better capture such linguistic features. Consequently, we extract part of speech tags using NLTK. Given the limited vocabulary of TinyStories, capabilities of NLTK POS tagger should be good enough for our purposes. Initially, there were more than 30 unique tags in the dataset, but many of them were very rare. After filtering out all tags with less than 200 occurrences, the following tags remained: CD — cardinal digit; IN — preposition or subordinating conjunction; JJ — adjective; NN - singular noun; NNP – proper noun; NNS – plural noun; RB – adverb; VB – base form verb; VBD - past tense verb; VBG - gerund; VBN - past participle. We use this notation in Table 3.

We added a feature indicating the frequency of the token in the training corpus because typically the direction with the most variance in the embedding space roughly corresponded to frequency. We also added another boolean feature that is one if the token starts with a whitespace and zero otherwise.

For each of the models and each of the features, we trained a ridge regression (for frequency) or a logistic regression (for all other variables, as they are Boolean) on 80% of the embeddings and then evaluated their R^2 score or ROC-AUC on the remaining 20%. See Table 3 for the results.

All probes in all models perform better than random, so every model learns at least something related to these word features. The embeddings of the model trained on English from scratch predictably outperformed the others, but the quality of other embeddings turned out to be on average the same. Perhaps the difference in effective dimension between the models is used not for these relatively simple single-word features, but for more complex ones.

6 Conclusion

We introduced a new synthetic language `flat_shuffle`, and the model pre-trained on it was shown to outperform the models based on the languages from previous work.

Investigation of the structure of the embeddings leads to a hypothesis that the reason behind the superior performance of some synthetic languages is that they require more structured embeddings, which causes the intermediate layers to be adapted to work with such embeddings, and in turn allows effectively using a higher dimension subspace of the embedding space during fine-tuning, which gives more flexibility.

Also, we haven't observed direct transfer of structure from synthetic languages to English, i.e. English tokens weren't splitted by the model into "opening" and "closing" ones. So it seems that models are working in a reservoir computing style where the computations for an unrelated task are adapted to the task at hand in arbitrary ways. At the same time, it means that the models are not strictly limited by the complexity or structure of the original task in transfer learning, and as long as they have enough complexity of computations, they can use it to adapt to the new task.

7 Limitations

The experiments reveal several interesting patterns about transfer learning between synthetic and natural languages. However, our approach has some important limitations.

First, we only used English as the target natural language. It would be interesting to see if the patterns we observed hold for other natural languages, especially those with different grammatical structures.

Second, even our most complex synthetic language, `flat_shuffle`, was simple enough to be learned by a model with 8 million parameters. Perhaps with better synthetic data and correspondingly more capable models we would observe qualitatively new phenomena.

Ethics Statement

This paper complies with the ACL Ethics Policy.

	nested	flat	flat_shuffle	scratch
frequency	0.84	0.85	0.85	0.93
start_space	0.70	0.70	0.70	0.89
pos_tag_CD	0.66	0.63	0.63	0.80
pos_tag_IN	0.76	0.79	0.71	0.87
pos_tag_JJ	0.60	0.58	0.60	0.73
pos_tag_NN	0.63	0.62	0.63	0.76
pos_tag_NNP	0.64	0.65	0.63	0.79
pos_tag_NNS	0.67	0.67	0.68	0.84
pos_tag_RB	0.69	0.63	0.64	0.84
pos_tag_VB	0.71	0.69	0.68	0.79
pos_tag_VBD	0.75	0.71	0.67	0.89
pos_tag_VBG	0.71	0.70	0.73	0.89
pos_tag_VBN	0.72	0.68	0.72	0.87
Average	0.70	0.68	0.68	0.84

Table 3: ROC-AUC for the linear probes on embeddings from models fine-tuned on English using embedding-only (E) mode.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2019. On the cross-lingual transferability of monolingual representations. *arXiv preprint arXiv:1910.11856*.

Mikhail Budnikov, Anna Bykova, and Ivan P Yamshchikov. 2024. Generalization potential of large language models. *Neural Computing and Applications*, pages 1–25.

Cheng-Han Chiang and Hung-yi Lee. 2022. On the transferability of pre-trained language models: A study from artificial datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10518–10525.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Xu Han, Zhengyan Zhang, Ning Ding, Yuxian Gu, Xiao Liu, Yuqi Huo, Jiezhong Qiu, Yuan Yao, Ao Zhang, Liang Zhang, et al. 2021. Pre-trained models: Past, present and future. *AI Open*, 2:225–250.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Philip A Huebner, Elior Sulem, Fisher Cynthia, and Dan Roth. 2021. Babyberta: Learning more grammar with small-scale child-directed language. In *Proceedings of the 25th conference on computational natural language learning*, pages 624–646.

Kundan Krishna, Jeffrey Bigham, and Zachary C Lipton. 2021. Does pretraining for summarization require knowledge transfer? *arXiv preprint arXiv:2109.04953*.

Matthias Lindemann, Alexander Koller, and Ivan Titov. 2023. Injecting a structural inductive bias into a seq2seq model by simulation. *arXiv preprint arXiv:2310.00796*.

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

Kevin Lu, Aditya Grover, Pieter Abbeel, and Igor Mordatch. 2021. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 1.

R Thomas McCoy, Erin Grant, Paul Smolensky, Thomas L Griffiths, and Tal Linzen. 2020. Universal linguistic inductive biases via meta-learning. *arXiv preprint arXiv:2006.16324*.

Sanket Vaibhav Mehta, Darshan Patil, Sarath Chandar, and Emma Strubell. 2021. An empirical investigation of the role of pre-training in lifelong learning. *arXiv preprint arXiv:2112.09153*.

Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. 2023. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40.

Behnam Neyshabur, Hanie Sedghi, and Chiyuan Zhang. 2020. What is being transferred in transfer learning? *Advances in neural information processing systems*, 33:512–523.

OpenAI. 2023. Gpt-4 technical report.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning music helps you read: Using transfer to study linguistic structure in language models. *arXiv preprint arXiv:2004.14601*.

Isabel Papadimitriou and Dan Jurafsky. 2023. Injecting structural hints: Using language models to study inductive biases in language learning.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Ryokan Ri and Yoshimasa Tsuruoka. 2022. Pretraining with artificial language: Studying transferable knowledge in language models. *arXiv preprint arXiv:2203.10326*.

Koustuv Sinha, Robin Jia, Dieuwke Hupkes, Joelle Pineau, Adina Williams, and Douwe Kiela. 2021. Masked language modeling and the distributional hypothesis: Order word matters pre-training for little. *arXiv preprint arXiv:2104.06644*.

Maxim K Surkov and Ivan P Yamshchikov. 2024. Vygotsky distance: measure for benchmark task similarity. *arXiv preprint arXiv:2402.14890*.

Pablo Villalobos, Jaime Sevilla, Lennart Heim, Tamay Besiroglu, Marius Hobbhahn, and Anson Ho. 2022. Will we run out of data? an analysis of the limits of scaling datasets in machine learning. *arXiv preprint arXiv:2211.04325*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Yuhuai Wu, Markus N Rabe, Wenda Li, Jimmy Ba, Roger B Grosse, and Christian Szegedy. 2021. Lime: Learning inductive bias for primitives of mathematical reasoning. In *International Conference on Machine Learning*, pages 11251–11262. PMLR.

A Appendix

Here are the examples of the Tiny Cloze benchmark for particular tasks. One example for each task:

- 'synonyms and antonyms':
  `"The box was incredibly light, almost as if it were #.", "empty", "full"`
- 'single plural':
  `They # to the store every Saturday.", "go", "goes"`
- 'logical relations':
  `"The dog barked loudly, # everyone woke up", "and", "but"`
- 'subject-verb agreement':
  `"The dog in the yard # every morning", "barks", "bark"`
- 'prepositions':
  `"The cat is sleeping # the chair","under","above"`
- 'conjunctions':
  `"She went to the store # she needed milk.", "because", "although"`
- 'temporal understanding':
  `"It's usually dark outside # the sun rises", "before", "while"`
- 'spatial understanding':
  `"The cat is under the #.", "table", "sky"`
- 'quantitative reasoning':
  `"There are 5 apples. If I eat 2, there will be # left", "3", "4"`
- 'emotions':
  `"When he lost his keys, he was really #.", "frustrated", "excited"`
- 'narrative understanding':
  `"After the long journey, the traveler was # and fell asleep quickly.", "tired", "hungry"`
- 'ethics':
  `"Cheating to win a game is # acceptable", "never", "always"`