Learning Music Helps You Read: Using Transfer to Study Linguistic Structure in Language Models

Isabel Papadimitriou
Stanford University
isabelvp@stanford.edu

Dan Jurafsky
Stanford University
jurafsky@stanford.edu

Abstract

We propose transfer learning as a method for analyzing the encoding of grammatical structure in neural language models. We train LSTMs on non-linguistic data and evaluate their performance on natural language to assess *which kinds of data induce generalizable structural features* that LSTMs can use for natural language. We find that training on non-linguistic data with latent structure (MIDI music or Java code) improves test performance on natural language, despite no overlap in surface form or vocabulary. To pinpoint the kinds of abstract structure that models may be encoding to lead to this improvement, we run similar experiments with two artificial parentheses languages: one which has a hierarchical recursive structure, and a control which has paired tokens but no recursion. Surprisingly, training a model on either of these artificial languages leads to the same substantial gains when testing on natural language. Further experiments on transfer between natural languages controlling for vocabulary overlap show that zero-shot performance on a test language is highly correlated with typological syntactic similarity to the training language, suggesting that representations induced by pre-training correspond to the cross-linguistic syntactic properties. Our results provide insights into the ways that neural models represent abstract syntactic structure, and also about the kind of structural inductive biases which allow for natural language acquisition. [1]

1 Introduction

Understanding how neural language models learn and represent syntactic structure is an important analytic question for NLP. Recent work has directly probed the internal activations of models (Conneau et al., 2018a; Dalvi et al., 2019; Hewitt and Manning, 2019; Clark et al., 2019), or fed them curated inputs that depend on complex syntax (Linzen et al., 2016; Gulordava et al., 2018; Talmor et al., 2019; McCoy et al., 2020), in order to uncover latent syntactic awareness.

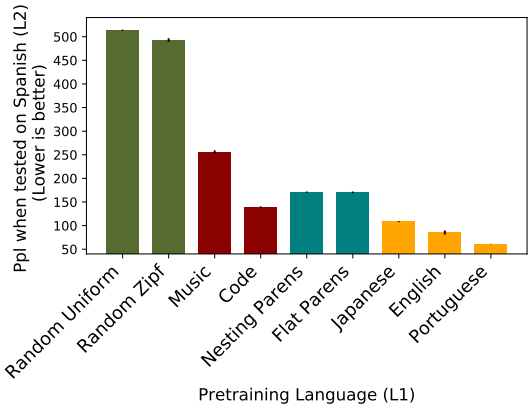


Figure 1: We find that LSTM LMs can utilize various types of non-linguistic structure to help learn to model human language, and that nested hierarchical structure does not lead to more expressive encodings than flat, head-dependency pair structure. We also find that LSTM LMs learn representations that correlate with typological syntactic feature distance, allowing them to transfer more effectively from languages which are grammatically similar.

We propose a different approach: we measure the structural awareness of a language model *by studying how much this structure acts as an inductive bias to improve learning when we transfer from one language or symbolic system to another.*

We train LSTM models on data with varying degrees of language-like structure (music, Java code, nested symbols), and then evaluate their performance on natural language. Before evaluation, we freeze the LSTM parameters and fine-tune the word embeddings on the evaluation language. This lets us see if the training data induces language-like

[1] We release code to construct the corpora and run our experiments at https://github.com/toizzy/tilt-transfer

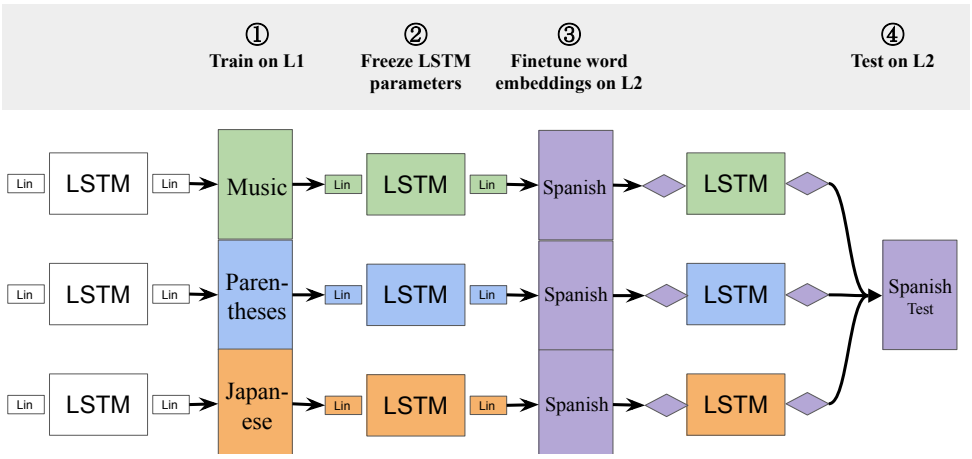


Figure 2: Diagram illustrating our training procedure: k models are trained on k L1 languages, and then their LSTM weights are frozen while their linear layers are finetuned on a common L2 language (in our case, we always use Spanish as the L2). We can then compare their performance on the common L2.

structure in the recurrent parameters of LSTMs—despite removing vocabulary-level confounders. By assessing if representations are useful *across* languages, we examine the generalizable representations of grammar that LSTMs encode. We call this new method the Test for Inductive Bias via Language Model Transfer (TILT).

Firstly, we examine the transfer of abstract structural features from languages that are very different on the surface from human language. We find that pretraining an LSTM on music data[2] or Java code greatly improves transfer to human language over pretraining on structureless random baseline data. To test if the gain in performance is due to the LSTM utilizing the recursive nature of music and code, we train models on an artificial language with recursion (hierarchically nested symbols) and observe that they also perform well when evaluated on human language. However, we also surprisingly find that recursion is a sufficient, but not necessary condition for generalizable, language-like grammar induction. We observe similar gains when pretraining on a language of matching pairs that do not nest hierarchically, showcasing the importance of non-hierarchical head-dependent-type relations in LSTM language processing.

Lastly, in transfer experiments between different human languages, we find that transfer is better between languages that are syntactically typologically similar, even with no vocabulary overlap. This suggests that models have the ability to form representations of typologically sensible properties rather than relying on ad-hoc or non-natural representations. For this result we draw on recent interlingual work such as Artetxe et al. (2020), Ponti et al. (2019), and Conneau et al. (2018b), extending it to use typological distance to turn these observations into quantitative probes.

The TILT method allows us to ask a complementary set of questions to those answered by current analysis methods. TILTs demonstrate the abstract structural notions that LSTMs can learn, rather than probing for the manifestation of a particular known structure, as in most current methods. By examining the pretraining structures that give LSTMs a better ability to model language, we also contribute to the more general cognitive question of what structural inductive biases a learner needs to be able to easily acquire human language.

2 Architecture and Training

Our methodology consists of training LSTM language models on k different first languages (L1s) which include natural languages, artificial languages, and non-linguistic symbol systems, and testing the performance of these models on a common second (L2) language. In our case, we used Spanish as the common L2. Before testing on the L2 test set, we fine-tune the linear embedding layer of the models on the L2 training set, while keeping the LSTM weights frozen. This aligns the vocabulary of each model to the new language, but does not let it learn any structural information about the L2 language. Though word embeddings do contain some grammatical information like part of

[2]We use the MAESTRO music dataset, which utilizes an exact symbolic representation of music (like a music score) that is sequentialized for sequence modelling

speech, they do not contain information about how to connect tokens to each other – that information is only captured in the LSTM. Figure 2 illustrates our training process. [3]

We vary the L1 languages and maintain a common L2 (instead of the other way around) in order to have a common basis for comparison: all of the models are tested on the same L2 test set, and therefore we can compare the perplexity scores. We run $n = 5$ trials of every experiment with different random seeds. Any high-resource human language would have provided a good common L2, and Spanish works well for our human languages experiments due to the fact that many higher-resource languages fall on a smooth gradation of typological distance from it (see Table 1).

We use the AWD-LM model (Merity et al., 2018) with the default parameters of 3 LSTM layers, 300-dimensional word embeddings, a hidden size of 1,150 per layer, dropout of 0.65 for the word embedding matrices and dropout of 0.3 for the LSTM parameters. We used SGD and trained to convergence, starting the learning rate at the default of 30 and reducing it at loss plateau 5 times.

Much of the work on multilingual transfer learning has speculated that successes in the field may be due to vocabulary overlap (see for example Wu and Dredze (2019)). Since our work focuses mostly on syntax, we wanted to remove this possibility. As such, we shuffle each word-to-index mapping to use disjoint vocabularies for all languages: the English word "Chile" and the Spanish word "Chile" would map to different integers. This addresses the confound of vocabulary overlap, as all language pairs have zero words in common from the point of view of the model.

Since the vocabularies are totally separated between languages, we align the vocabularies for all L1-L2 pairs by finetuning the word embeddings of all the pretrained models on the Spanish (L2) training data, keeping the LSTM weights frozen. By doing this, we remove the confound that would arise should one language's vocabulary randomly happen to be more aligned with Spanish than another's. These controls ensure that lexical features, whether they be shared vocabulary or alignment of randomly aligned indices, do not interfere with the experimental results which are meant to compare higher-level syntactic awareness.

[3] All pretraining jobs took less than 2 days to run on one GPU, all finetuning jobs took less than 1 day to run on one GPU.

3 Experiment 1: Random Baselines

We run our method on a random baseline L1: a corpus where words are sampled uniformly at random. This gives us a baseline for how much information we gain finetuning the word embeddings to the L2, when there has not been any structurally biasing input to the LSTM from the L1.

We also examine the importance of vocabulary distribution by training on a random corpus that is sampled from a Zipfian distribution. Human languages are surprisingly consistent in sharing a roughly Zipfian vocabulary distribution, and we test how pretraining on this distribution affects the ability to model human language. [4]

3.1 Data

Our random corpora are sampled from the Spanish vocabulary, since Spanish is the common L2 language across all experiments. Words are sampled uniformly for the Uniform Random corpus, and drawn from the empirical Spanish unigram distribution (as calculated from our Spanish training corpus) for the Zipfian Random corpus. Illustrative examples from all of our corpora can be found in Figure 3. The random corpora are controlled to 100 million tokens in length.

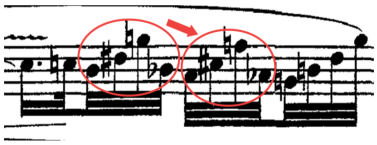
3.2 Results

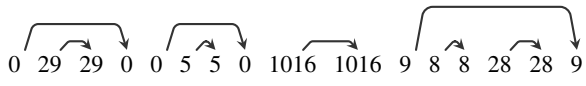
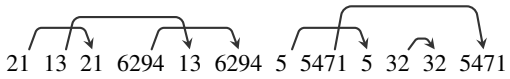
When tested on Spanish, the average perplexity is 513.66 for models trained on the Random Uniform corpus and 493.15 for those trained on the Random Zipfian corpus, as shown in Figure 4. These perplexity values are both smaller than the vocabulary size, which indicates that the word embedding finetuning captures information about the test language even when the LSTM has not been trained on any useful data.

The models trained on the Zipfian Random corpus are significantly better than those trained on the Uniform corpus ($p << 0.05$, Welch's t-test over $n = 5$ trials). However, even though training on a Zipfian corpus provides gains when compared to training on uniformly random data, in absolute terms performance is very low. This indicates that, without higher-level language-like features, there is very little that an LSTM can extract from properties of the vocabulary distribution alone.

[4] See Piantadosi (2014) for a review of cognitive, communication and memory-based theories seeking to explain the ubiquity of power law distributions in language.

<table>
<tr><td><u>Random</u>

Uniform: <code>marroquín jemer pertenecer osasuna formaron citoesqueleto relativismo</code></td><td rowspan="2">The random corpora are sampled randomly from the Spanish vocabulary. There is no underlying structure of any kind that links words with each other. All words are equally likely to be sampled in the Uniform corpus, while common words are more likely in the Zipfian corpus.</td></tr>
<tr><td>Zipf: <code>en con conocidas y en los victoriano como trabajar ⟨unk⟩ monte * en juegos días en el</code></td></tr>
<tr><td><u>Music</u>
  </td><td>The music data is encoded from classical piano performances according to the MAESTRO standard. Music is structured on many levels. The red arrow in the example illustrates how, on a small timescale, each note is linked to its corresponding note when a motif is repeated but modulated down a whole-step.</td></tr>
<tr><td><u>Code</u>
<pre>if (coordFactor == 1.0f)
    return sumExpl
else {
    result = sum * coordFactor
}</pre></td><td>The code corpus is composed of Java code. The above snippet demonstrates some kinds of structure that are present in code: brackets are linked to their pairs, <code>else</code> statements are linked to an <code>if</code> statement, and coreference of variable names is unambiguous.</td></tr>
<tr><td><u>Parentheses</u>

Nesting:
 
Flat:
  </td><td>Our artificial corpora consist of pairs of matching integers. In the Nesting Parentheses corpus, integer pairs nest hierarchically and so the arcs do not cross. In the Flat Parentheses corpus, each integer pair is placed independently of all the others, and so the arcs can cross multiple times.

(There is a one-to-one mapping between Spanish words and integers and so these integers are sampled from the same Spanish vocabulary distribution as the Random Zipfian corpus. We visualize these corpora here with integers and the Random corpora with words for simplicity).</td></tr>
</table>

Figure 3: Examples illustrating the content of our non-linguistic corpora for Experiments 1-3. All examples are taken from the corpora.

The Zipfian Random baseline is controlled for vocabulary distribution: if an experiment yields better results than the Zipfian Random baseline, we cannot attribute its success only to lexical-level similarity to the L2. Therefore, models that are more successful than the Zipfian baseline at transfer to human language would have useful, generalizable syntactic information about the structures that link tokens.

4 Experiment 2: Non-linguistic structure

In this experiment, we test the performance of LSTMs on Spanish when they have been trained on music and on code data. While music data especially is very different from human language on the surface level, we know that music and code both contain syntactic elements that are similar to human language.[5] By comparing performance to our random baselines, we ask: can LSTMs encode the abstract structural features that these corpora share with natural language in a generalizable way that's usable to model human language?

4.1 Data

For our music data we use the MAESTRO dataset of Hawthorne et al. (2018). The MAESTRO dataset embeds MIDI files of many parallel notes into a linear format suitable for sequence modelling, without losing musical information. The final corpus has a vocabulary of 310 tokens, and encodes over 172 hours of classical piano performances. [6]

For programming code data, we used the Habeas corpus released by Movshovitz-Attias and Cohen (2013), of tokenized and labelled Java code. [7] We took out every token that was labelled as a comment so as to not contaminate the code corpus with

[5] See for example Lerdahl and Jackendoff (1996) for grammatical structure in music.

[6] The MAESTRO dataset is available at https://magenta.tensorflow.org/datasets/maestro

[7] The Habeas corpus is available at https://github.com/habeascorpus/habeascorpus-data-withComments

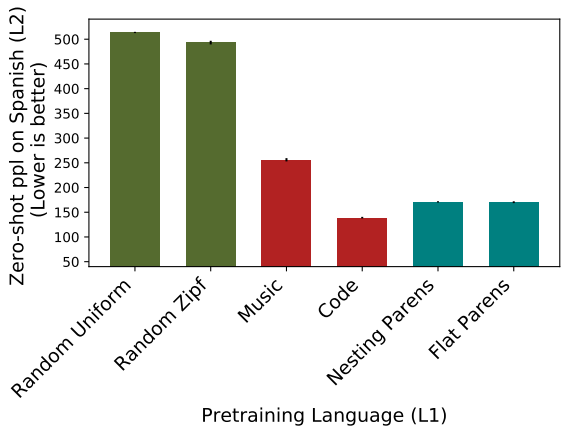


Figure 4: Results of Experiments 1 through 3, training on non-linguistic corpora. Error bars on all bars indicate a 95% t-test confidence interval over 5 restarts with different random seeds. All structured data is much better to train on than random data, including music which has a totally divergent vocabulary surface form from the rest. The two parentheses corpora result in equivalent perplexities, even though one has a hierarchical underlying structure and the other does not.

natural language.

The music corpus is 23 million tokens in length and the code corpus is 9.5 million. We cannot effectively control the lengths of these corpora to be the same as all of the others, since there is no controlled notion of what one token means in terms of information. However, we only compare these results to the random baseline, which we have trained on 100 million tokens – if the LSTMs trained on these corpora are under-specified compared to the baseline, this would only strengthen our results.

4.2 Results

Our results show that language models pretrained on music are far better at modelling Spanish than those pretrained on random data. As shown in figure 4, LSTMs trained on music data have an average performance of 256.15 ppl on Spanish, compared with 493.15 when training on the Zipfian random corpus. This discrepancy suggests that the model, when training on music, creates representations of the relationships between tokens which are generalizable and can apply to Spanish.

The music corpus is markedly different from the Spanish corpus by most measures. Most saliently, MAESTRO uses a vocabulary of just 310 tokens to encode various aspects of music like volume and note co-occurrence.[8] This is in contrast to the Zipfian Random corpus, which has the same surface-level vocabulary and distribution as Spanish, yet models trained on it perform on average 237 ppl worse compared to those trained on the music corpus. Since the surface forms between music and language are so different, the difference in performance cannot be based on surface-level heuristics, and *our results suggest the presence of generalizable, structurally-informed representations in LSTM language models*.

We also show that models trained on Java code can transfer this knowledge to a human L2 better than the random baseline. Syntactic properties of code such as recursion are similar to natural language, though code is constructed to be unambiguously parsed and lacks a lot of the subtlety and ambiguity that characterizes natural language. Models trained on code have an average perplexity of 139.10 on the Spanish test set. The large discrepancy between this performance and the baseline indicates that LSTMs trained on code capture the syntactic commonalities between code and natural language in a manner that is usable for modelling natural language.

Our results on non-linguistic data suggest that LSTMs trained on structured data extract representations which can be used to model human languages. The non-linguistic nature of these data suggests that it is something structural about the music and Java code that is helping in the zero-shot task. However, there is a multitude of structural interpretations of music, and it is not clear what kinds of structure the LSTM encodes from music. In the next experiment, we create simple artificial corpora with known underlying structures in order to test how the LMs can represent and utilize these structures.

5 Experiment 3: Recursive Structure

In this experiment, we isolate and assess possible structural features of music and code that may explain the results of Experiment 2. The most widely-known structural hypothesis is the claim of Hauser et al. (2002) that the narrow language faculty in humans (the inductive bias in the mind/brain that allows humans to acquire and develop language) can be reduced to just recursion. Given the prominence of such theories, it is natural to ask: is it the

[8] For consistency, the model still has a word embedding matrix of 50,000 rows, but during training only ever sees words 1-310, meaning that much of the word embedding space has never been seen by the LSTM part of the model.

underlying recursive nature of music and code data that causes the gains that we observe in Experiment 2?

To test this possibility, we create a simple recursive corpus: a *Nesting Parentheses corpus* of hierarchically nesting matching symbols, and run the same experimental setup as we did for Experiments 1 and 2 [9]. We find that plain recursion, even when the corpus has no other structural subtleties, is indeed a sufficient condition for inducing the kinds of structural transfer we observed in Experiment 2.

Recursion is a sufficient quality, but is it the only explanation for our results? We also create a control corpus: a *Flat Parentheses corpus*, which has similar pairs of matching parentheses, but which do not nest hierarchically and projectively (the difference between the two corpora is visually illustrated in Figure 3). We surprisingly find that this non-recursive corpus induces the same amount of structural transfer as the recursive nesting parentheses, which emphasizes the importance of pairing, head-dependency type structure in the linguistic structural embeddings of LSTMs.

5.1 Data

The vocabulary for these corpora are the integers 0-50,000, where each number is a parenthesis token, and that token "closes" when the same integer appears a second time. We draw the opening tokens from the empirical Spanish unigram distribution (mapping each Spanish word to an integer), meaning that these corpora have a similar vocabulary distribution, albeit a much simpler non-linguistic structure, to the L2. Both of the corpora are 100 million tokens long, like the random and the natural language corpora.

We create the Nesting Parentheses corpus by following a simple stack-based grammar. At timestep t, we flip a coin to decide whether to open a new parenthesis (with probability 0.4) or close the top parenthesis on the stack (with probability 0.6).[10] If we are opening a new parenthesis, we sample an integer x_{open} from the Spanish unigram distribution, write the integer x_{open} at the corpus position t, and push x_{open} onto the stack of open parentheses. If we are closing a parenthesis, we pop the top integer from the stack, x_{close}, and write x_{close} at corpus position t.

The Flat Parentheses corpus is made up of pairs of parentheses that do not nest. At timestep t, we sample an integer x from the empirical Spanish unigram distribution, and a distance d from the empirical distribution of dependency lengths (calculated from the Spanish Universal Dependencies treebank (McDonald et al., 2013)). Then, we write x at position t and at position $t + d$. This creates pairs of matching parentheses which are not influenced by any other token in determining when they close. *Note that this corpus is very similar to the Random Zipf corpus, except that each sampled token is placed twice instead of once.*

5.2 Results

LSTMs trained on both parentheses corpora are able to model human language far better than models trained on the random corpora, indicating that the isolated forms of grammar-like structure in these corpora are useful for modelling human language. Surprisingly, performance is the same for a model pretrained on the Nesting Parentheses and the Flat Parentheses corpus. This suggests that it is not necessarily hierarchical encodings which LSTMs use to model human language, and that other forms of structure such as flat head-head dependencies may be just as important (de Marneffe and Nivre, 2019).

The Nesting Parentheses corpus exhibits hierarchical structure while not having any of the irregularities and subtleties of human language or music. Despite the simplicity of the grammar, our results indicate that the presence of this hierarchical structure is very helpful for an LSTM attempting to model Spanish. Our models trained on the Nesting Parentheses corpus have an average perplexity of 170.98 when tested on the Spanish corpus. This is 322 perplexity points better than the baseline models trained on the Zipf Random corpus, which has the same vocabulary distribution (Figure 4).

Models trained on the Flat Parentheses corpus are equally effective when tested on Spanish, achieving an average perplexity of 170.03. These results are surprising, especially given that the Flat Parentheses corpus is so similar to the Random Zipf corpus – the only difference being that integers are placed in pairs not one by one – and yet performs better by an average of 323 perplexity

[9] Though these corpora do not strictly use parentheses tokens, we refer to both of these as parentheses corpora, drawing our metaphor from the wide variety of studies such as Karpathy et al. (2016) examining nested parentheses.

[10] $P(open)$ has to be strictly less than 0.5, or else the tree depth is expected to grow infinitely.

Language	WALS-syntax distance from Spanish (out of a max of 49 features)
Spanish (es)	0
Italian (it)	0
Portuguese (pt)	3
English (en)	4
Romanian (ro)	5
Russian (ru)	9
German (de)	10
Finnish (fi)	13
Basque (eu)	15
Korean (ko)	18
Turkish (tr)	23
Japanese (ja)	23

Table 1: WALS-syntax distance between Spanish and L1s

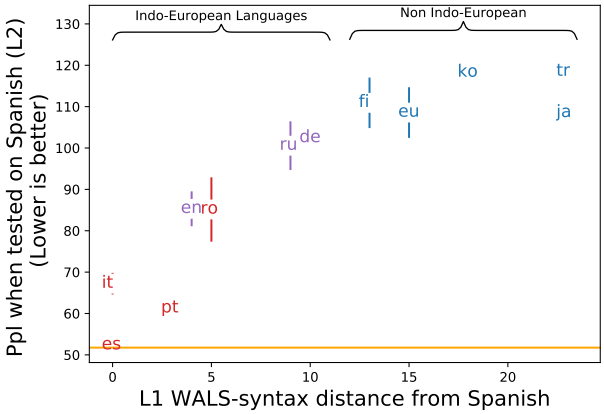


Figure 5: Results of Experiment 4. Transfer is better between typologically similar languages, even when vocabularies are disjoint. Perplexity on Spanish test data plotted against the WALS-syntax distance of each model's L1 to Spanish. The relationship is almost linear for Indo-European languages, and then reaches a ceiling. Error bars show 95% CIs for $n = 5$ trials with different random seeds. These results demonstrate how LSTMs can transfer knowledge more easily to languages that share structural features with the L1, and that this correlation is robust to multiple trials. The orange line represents the oracle perplexity of training all parameters to convergence on the L2 train data. Romance languages are in red, other Indo-European languages are in purple, and non-Indo-European languages are blue.

points. This suggests that *representing relationships between pairs of tokens is a key element that makes syntactic representations of language successful in LSTMs*.

The Flat Parentheses corpus has structure in that each token is placed in relation to one other token, but just one other token. To model this successfully a model would have to have some ability to look back at previous tokens and determine which ones would likely have their match appear next. Our results suggest that this kind of ability is just as useful as potentially being able to model a simple stack-based grammar.

6 Experiment 4: Human Languages

To further analyze what kinds of generalizable structure LSTMs can infer, we run experiments in transferring zero-shot between human languages. We ask: can LSTMs infer and use fine-grained syntactic similarities between typologically similar languages? Previous work (Zoph et al., 2016; Artetxe et al., 2020) indicates that transfer is more successful between related languages. We control for vocabulary overlap, and use typological syntactic difference as a quantitative probe to ask: are fine-grained syntactic similarities encoded in generalizable, transferrable ways? To answer this question, we investigate the extent to which fine-grained differences in syntactic structure cause different zero-shot transfer results.

6.1 Data

We created our language corpora from Wikipedia, which offers both wide language variation as well as a generally consistent tone and subject domain. We used the gensim wikicorpus library to strip Wikipedia formatting, and the stanfordnlp Python library (Qi et al., 2018) to tokenize the corpus. We run experiments on data from 12 human languages, all of which have Wikipedias of over 100,000 articles: Spanish, Portuguese, Italian, Romanian, English, Russian, German, Finnish, Basque, Korean, Turkish and Japanese. All of the training corpora are 100 million tokens in length. [11]

For our typological data, we use the World Atlas of Linguistic Structure, using the features that relate to syntax (WALS-syntax features). Examples of syntactic features in WALS include questions such as does a language have Subject-Verb-Object order, or does a degree word (like "very") come before or after the adjective. We accessed the WALS data using the lang2vec package (Littell et al., 2017). The quantity we are interested in extracting from the WALS data is the *typological distance* between the L2 (Spanish) and all of the

[11] The code for recreating our corpora from Wikipedia dumps is available at https://github.com/toizzy/wiki-corpus-creator

L1 languages mentioned above. Not every feature is reported for every language, so we calculate the WALS distance by taking into account only the 49 (syntactic) features that are reported for all our chosen languages and count the number of entries that are different (see Table 1). Since they are only based on 49 features, these distances do not provide a perfectly accurate distance metric. Though we cannot use it for fine-grained analysis, correlation with this distance metric would imply correlation with syntactic distance.

6.2 Results

Our experiments present a strong correlation between the ability to transfer from an L1 language to Spanish and the WALS-syntax distance between those two languages, as shown in Figure 5(a). In the case of Indo-European languages the relationship is largely linear with a Pearson R^2 coefficient of 0.83. For languages not in the Indo-European language family, transfer performance appears to reach a noisy ceiling, and Pearson's $R^2 = 0.78$ when taking into account all languages.[12]

Our previous experiments show that LSTMs can encode and generalize structural features from data that is structured, both in recursive and in non-hierarchical fashion. This experiment provides a more fine-grained analysis using using natural language to show that the syntax induced by LSTMs is generalizable to other languages in a typologically sensible fashion, even when we do not let the model take advantage of vocabulary overlap. However, after a certain threshold, the model is unable to take advantage of fine-grained similarities and performance on distant languages reaches a ceiling. It should be noted that all of the models trained on natural language, even the most distant, perform far better than non-linguistic data, indicating that LSTMs are able to extract universal syntactic information from all natural language L1s that is applicable to Spanish.

7 Discussion

In this work we propose the Test for Inductive bias via Language model Transfer (TILT), a novel analytic method for neural language models which tests the ability of a model to *generalize and use structural knowledge*. We pretrain LSTMs on structured data, and then use the frozen LSTM weights to model human language. In doing so, we treat the frozen LSTM weights as the only structural faculty available to a human language model, and assess if the induced structure is general enough to be used to model human language.

Our experiments are cross-lingual and cross-modal in nature, not searching for representations of high-level features in one language, but for representations that encode general ideas of structure. While the majority of past work analyzing the structural abilities of neural models looks at a model's treatment of structural features that are realized in specific input sentences, our method compares the *encoding and transfer of general grammatical features of different languages*. By using TILTs, we do not have to identify a structural feature of interest and investigate if it is being encoded, but instead asses if generalizable abstract structures are encoded in one language by examining *if they can be used* to model human language. Our work thus avoids known issues that have been pointed out with analytic methods like probing (Voita and Titov, 2020; Pimentel et al., 2020; Hewitt and Liang, 2019).

We run experiments on natural languages, artificial languages, and non-linguistic corpora. Our non-linguistic and artificial language experiments suggest three facets of the structural encoding ability of LSTM LMs. First, that vocabulary distribution has a very minor effect for modelling human language compared to structural similarity. Second, that models can encode useful language modelling information from the latent structure inherent in non-linguistic structured data, even if the surface forms are vastly differing. Last, that encodings derived from hierarchically structured tokens are equally useful for modelling human language as those derived from texts made up of pairs of tokens that are linked but non-hierarchical. Running experiments on a range of human languages, we conclude that the internal linguistic representation of LSTM LMs allows them to take advantage of structural similarities between languages even when unaided by lexical overlap.

Our results on the parentheses corpora do not necessarily provide proof that the LSTMs trained on the Nesting Parentheses corpus aren't encoding and utilizing hierarchical structure. In fact, previous research shows that LSTMs are able to suc-

[12] We verified that our results also stand when calculating correlation coefficients using log perplexity, which yielded similar values: R^2 of 0.79 and 0.73 for Indo-European and all languages respectively.

cessfully model stack-based hierarchical languages (Suzgun et al., 2019b; Yu et al., 2019; Suzgun et al., 2019a). What our results do indicate is that, in order for LSTMs to model human language, being able to model hierarchical structure is similar in utility to having access to a non-hierarchical ability to "look back" at one relevant dependency. These results shine light on the importance of considering other types of structural awareness that may be used by neural natural language models, even if those same models also demonstrate the ability to model pure hierarchical structure.

Our method could be used to test many other hypotheses regarding neural language models, by choosing a discerning set of pretraining languages. A first step in future work would be to test if the results of this paper hold on Transformer architectures, or if instead Transformers result in different patterns of structural encoding transfer. Future work expanding on our results could focus on ablating specific structural features by creating hypothetical languages that differ in single grammatical features from the L2, in the style of Galactic Dependencies (Wang and Eisner, 2016), and testing the effect of structured data that's completely unrelated to language, such as images.

Our results also contribute to the long-running nature-nurture debate in language acquisition: whether the success of neural models implies that unbiased learners can learn natural languages with enough data, or whether human abilities to acquire language given sparse stimulus implies a strong innate human learning bias (Linzen and Baroni, 2020). The results of our parentheses experiments suggest that simple structural head-dependent bias, which need not be hierarchical, goes a long way toward making language acquisition possible for neural networks, highlighting the possibility of a less central role for recursion in language learning for both humans and machines.

Acknowledgements

We thank Urvashi Khandelwal, Kawin Ethayarajh, Kyle Mahowald, Chris Donahue, Yiwei Luo, Alex Tamkin and Kevin Clark for helpful discussions and comments on drafts, and our anonymous reviewers for their feedback. This work was supported by an NSF Graduate Research Fellowship for IP and a SAIL-Toyota Research Award. Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018a. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018b. XNLI: Evaluating cross-lingual sentence representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium. Association for Computational Linguistics.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James Glass. 2019. What is one grain of sand in the desert? analyzing individual neurons in deep NLP models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6309–6317.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Marc D. Hauser, Noam Chomsky, and W. Tecumseh Fitch. 2002. The faculty of language: What is it, who has it, and how did it evolve? *Science*, 298(5598):1569–1579.

Curtis Hawthorne, Andriy Stasyuk, Adam Roberts, Ian Simon, Cheng-Zhi Anna Huang, Sander Dieleman, Erich Elsen, Jesse Engel, and Douglas Eck. 2018. Enabling factorized piano music modeling and generation with the maestro dataset.

John Hewitt and Percy Liang. 2019. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods*

in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 2733–2743.

John Hewitt and Christopher D Manning. 2019. A structural probe for finding syntax in word representations. In *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)*, pages 4129–4138.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2016. Visualizing and understanding recurrent networks.

Fred Lerdahl and Ray S Jackendoff. 1996. *A generative theory of tonal music*. MIT press.

Tal Linzen and Marco Baroni. 2020. Syntactic structure from deep learning. *Annual Review of Linguistics*, 7.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Patrick Littell, David R Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. Uriel and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 8–14.

Marie-Catherine de Marneffe and Joakim Nivre. 2019. Dependency grammar. *Annual Review of Linguistics*, 5:197–218.

R. Thomas McCoy, Robert Frank, and Tal Linzen. 2020. Does syntax need to grow on trees? sources of hierarchical inductive bias in sequence-to-sequence networks. *Transactions of the Association for Computational Linguistics*, 8:125–140.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, et al. 2013. Universal dependency annotation for multilingual parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 92–97.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2018. Regularizing and optimizing LSTM language models. In *International Conference on Learning Representations*.

Dana Movshovitz-Attias and William Cohen. 2013. Natural language models for predicting programming comments. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 35–40.

Steven T Piantadosi. 2014. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic bulletin & review*, 21(5):1112–1130.

Tiago Pimentel, Josef Valvoda, Rowan Hall Maudslay, Ran Zmigrod, Adina Williams, and Ryan Cotterell. 2020. Information-theoretic probing for linguistic structure.

Edoardo Maria Ponti, Ivan Vulić, Ryan Cotterell, Roi Reichart, and Anna Korhonen. 2019. Towards zero-shot language modeling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2900–2910, Hong Kong, China. Association for Computational Linguistics.

Peng Qi, Timothy Dozat, Yuhao Zhang, and Christopher D. Manning. 2018. Universal dependency parsing from scratch. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 160–170, Brussels, Belgium. Association for Computational Linguistics.

Mirac Suzgun, Sebastian Gehrmann, Yonatan Belinkov, and Stuart M. Shieber. 2019a. LSTM networks can perform dynamic counting. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, volume abs/1906.03648, Florence. Association for Computational Linguistics.

Mirac Suzgun, Sebastian Gehrmann, Yonatan Belinkov, and Stuart M. Shieber. 2019b. Memory-augmented recurrent neural networks can learn generalized Dyck languages.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. oLMpics – on what language model pre-training captures.

Elena Voita and Ivan Titov. 2020. Information-theoretic probing with minimum description length. *arXiv preprint arXiv:2003.12298*.

Dingquan Wang and Jason Eisner. 2016. The Galactic Dependencies treebanks: Getting more data by synthesizing new languages. *Transactions of the Association for Computational Linguistics*, 4:491–505.

Shijie Wu and Mark Dredze. 2019. Beto, bentz, becas: The surprising cross-lingual effectiveness of bert. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Xiang Yu, Ngoc Thang Vu, and Jonas Kuhn. 2019. Learning the dyck language with attention-based seq2seq models. In *ACL 2019*.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *Proceedings of the*

2016 Conference on Empirical Methods in Natural Language Processing, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

Appendix: Numerical results of experiments

For every experiment we ran five trials with different random seeds. We list the means and standard deviations for each L1 below:

L1 Language	Mean TILT Ppl	Std. Dev
Random Uniform	513.66	1.01
Random Zipf	493.15	2.97
Music	256.15	2.65
Code	139.11	1.24
Nesting Parens	170.98	1.02
Flat Parens	170.30	1.48
Basque	108.57	4.93
English	85.30	3.40
Finnish	110.92	3.84
German	102.42	0.51
Italian	67.21	2.10
Japanese	108.48	0.81
Korean	118.23	1.23
Portoguese	61.25	0.21
Romanian	85.14	6.26
Russian	100.56	4.74
Spanish	52.33	0.21
Turkish	118.45	0.85