XferBench: a Data-Driven Benchmark for Emergent Language

Anonymous ACL submission

Abstract

In this paper, we introduce a benchmark for evaluating the overall quality of emergent languages using data-driven methods. We define the notion of the "quality" of an emergent language as its similarity to human language within a deep learning framework. We measure this by using the emergent language as pretraining data for a downstream NLP tasks in human language—the better the downstream performance, the better the emergent language. We package this benchmark as an easy-to-use Python package that only requires a text file of utterances from the emergent language to be evaluated. Finally, we empirically test the benchmark's validity using human, synthetic, and emergent language baselines.

1 Introduction

Neural language models learn many things in pretraining, but research suggests (Artetxe et al., 2020) that a substantial part of that knowledge is not simply knowledge of a particular language or domain, but rather knowledge of "how to language." We currently teach models to "language" using vast quantities of text dredged from the dark recesses of the Web—text that is full of bias, toxicity, and potential intellectual property violations. Ideally, we would be able to teach models to "language" without such compromises through the use of synthetic data, but mainstream approaches to synthesizing data produce outputs that do not have the same structural and social properties as human language.

Emergent communication (EC), also called emergent language (EL), is a potential solution to this problem (Yao et al., 2022; Downey et al., 2023; Mu et al., 2023). Emergent languages are communication systems developed among multiple agents in a reinforcement learning simulation. Because the conditions under which they develop mirror, reductively, the conditions under which languages develop among humans, there is reason to believe that ELs will ultimately be more like human language than other sources of synthetic data. However, up to this point, there is no way of quantifying—in a holistic way—how much like human languages any particular EL really is, or to what extent it may provide useful pretraining signals.

Research on deep learning-based emergent communication has seen the introduction of many metrics to measure various aspects of the language. These metrics quantify notions such as compositionality (Brighton and Kirby, 2006; Lazaridou et al., 2018), expressivity (Guo et al., 2023), ease-of-teaching Li and Bowling (2019), and zero-shot transfer Bullard et al. (2020), to name a few. Despite this proliferation of metrics, emergent language largely lacks *evaluation* metrics. An evaluation metric is specifically one that measures the *overall quality of an emergent language* and not simply a particular property. Thus, we introduce XferBench, a data-driven benchmark for evaluating the overall quality of emergent languages using transfer learning with deep neural models.

Evaluation metrics are critical in gauging progress in technical fields since they quantify otherwise vague notions of improvement over time. Benchmarks, in particular, pair evaluation metrics with specific data and evaluation procedures to compare various models on common ground. Benchmarks and shared tasks have been critical to the development of NLP from the Penn Treebank (Marcus et al., 1993) to the WMT datasets (Bojar et al., 2014) to GLUE (Wang et al., 2018).

In the field of emergent communication specifically, Yao et al. (2022) introduced the idea of using *corpus transfer* as means of practically applying emergent communication to deep learning-based NLP via transfer learning. In corpus transfer, a language model is pretrained on a corpus of emergent language utterances before being tuned on real data for a human language-based downstream task. As a corollary, they suggest that the effectiveness of this transfer can serve as a means of evaluating the quality of the emergent in a more general sense. This is based on the intuition that the more similar two language are, the better transfer learning works from one to the other (observed in Zoph et al. (2016), for example).

This paper takes the transfer learning-as-an-evaluation metric idea from Yao et al. (2022) and expands it into a full benchmark, XferBench, for emergent languages (illustrated in Figure 1). An evaluation metric for emergent languages in a benchmark format is the first of its kind. Additionally, XferBench is unique within emergent communication for being primarily data-driven instead of relying on particular handcrafted algorithms for quantifying a given phenomenon. This means that XferBench can be easily scaled up in the

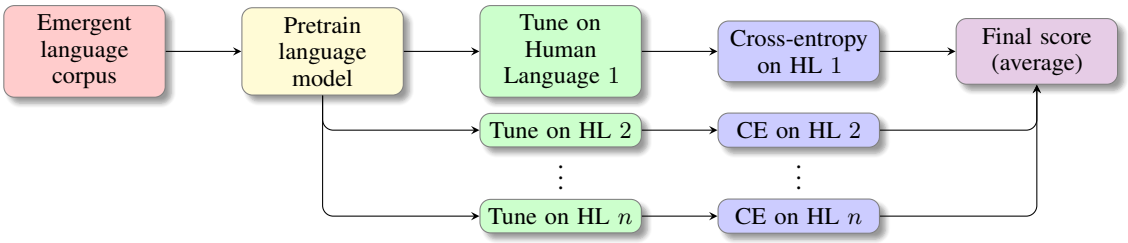

Figure 1: Illustration of the architecture of XferBench.

future as the field of emergent communication advances and requires expanded means of evaluating emergent languages. Finally, XferBench is distributed as a user-friendly Python package, allowing researchers from across the field of emergent communication to apply XferBench to their own work on emergent communication.

Contributions This paper makes the following contributions: (1) Introduces XferBench, a data-driven benchmark for evaluating the overall quality of an emergent language, the first of its kind in emergent communication. (2) Provides a analysis of the quality human, synthetic, and emergent language according to XferBench. (3) Provides an easy-to-use Python implementation of XferBench.

2 Related Work

Emergent Communication This paper is situated in the field of emergent communication (a.k.a. emergent language) which is generally covered by the review Lazaridou and Baroni (2020). The field centers around the invention of language by deep neural networks typically using multi-agent reinforcement learning techniques. The study of emergent communication is intended to (1) shed light on the origin and nature of the human language (LaCroix, 2019; Moulin-Frier and Oudeyer, 2020; Galke et al., 2022) and (2) provide an alternative approach to problems in NLP and multi-agent reinforcement learning which relies on constructing language from the ground up and not just pre-existing (human) languages alone (Li et al., 2020; Yao et al., 2022; Mu et al., 2023; Downey et al., 2023).

Transfer Learning Transfer learning for deep neural networks is a key component of XferBench and follows in general tradition of Zoph et al. (2016). Specifically, this paper draws heavily from Yao et al. (2022) (see also Papadimitriou and Jurafsky (2020); Artetxe et al. (2020)) which introduce the technique of *corpus transfer* for emergent language, that is, pretraining a neural model on an emergent language corpus before tuning it on a downstream human language task. In particular, this paper takes Yao et al. (2022)'s idea of using corpus transfer as a metric and adapts it into a benchmark pipeline which can easily be applied to new emergent languages.

Benchmarks Work such as Guo et al. (2023) and Perkins (2022) have looked at benchmarking particular aspects of emergent languages, but XferBench is the first of its kind in benchmarking the overall quality of an emergent language. Yao et al. (2022) also explicitly provide a metric for emergent language quality, but this metric is restrictive in that it can only be applied to emergent languages derived from a model that takes images (that have captions available) as input; this conflicts with the design goals of XferBench discussed below.

Outside of emergent communication, XferBench is more analogous to benchmarks for generative models (e.g., Fréchet Inception Distance (Heusel et al., 2017) for image generation) than more traditional NLP benchmarks like GLUE (Wang et al., 2018) or SQuAD (Rajpurkar et al., 2016). This is because emergent communication is a generative enterprise, where one of the main goals is to create samples (emergent languages) which resemble a target distribution (human languages) either generally or in some particular respect. Furthermore, metrics like FID are primarily self-supervised, data-driven measures of similarity along the same vein of XferBench. This is in contrast to more traditional NLP benchmarks which combine data-driven methods with many human judgments (i.e., through labeled examples).

3 XferBench

3.1 Design Goals

We frame the primary design goals of the benchmark as three desiderata:

- **D1** Quantitatively capture a meaningful notion of the overall quality of an emergent language from a data-driven perspective.
- **D2** Be applicable to any emergent language, not restricted to a specific game, environment, or agent architecture.
- **D3** Be relevant and accessible to the broader EC/EL community, by being: (a) easy to interpret, (b) minimally biased with regards to language typology, (c) runnable with minimal coding experience, and (d) runnable on modest hardware.

While there are other consideration in the benchmark, these form the bulk of the motivation. In the following paragraphs we expand upon the motivation for each design goal.

D1: Quantifying quality D1 is the core of what a benchmark seeks to do: to quantify a desirable property of a given system such that it can be compared directly to other systems (i.e., be an *evaluation* metric). There are two distinct senses in which XferBench strives towards this goal. First, XferBench measures how good an emergent language is from a specifically machine learning perspective; that is, it addresses the question, "How useful would this emergent language be for practical machine learning tasks?" The second sense is more general: XferBench addresses the question, "How similar is an emergent language to human language from the perspective of machine learning methods?" That is, it uses data-driven techniques to quantify the similarity between emergent language and human language.

D2: Wide applicability D2 is intended to make XferBench practically applicable to a wide range of EC research. The field of EC has an especially diverse set of possible approaches, environments, agents, games, etc. Thus, it is especially salient that the benchmark be designed with interoperability in mind, having minimal assumptions as to the nature of the EC system being evaluated.

The influence of this design goal is primarily seen through the use of a textual corpus as the sole input to the benchmark: the vast majority of EC systems generate utterances which can be represented as sequences of discrete tokens.[1] EC presents the opportunity for much richer representations of its language: leveraging the grounded semantics of the communication, incorporating non-verbal behavior, and even directly interacting with the agents themselves. Yet such richer representations also limit the range of EC systems to which XferBench could apply. Even if it is possible to define some universal EC interface that could allow for richer representations, the implementation cost for each and every EC system to be tested is significant compared to the ease of producing a corpus of utterances from the emergent language.

D3: Easy-to-use D3 is critical to the success of XferBench as a practical tool for diverse field of researchers—a benchmark is expressly *for* the broader research community, and, as such, should be widely usable. In particular, D3a demands that XferBench be conceptually simple with results that can easily be reported, compared, and incorporated into a research program. D3b is relevant to both aspects of D1. First, if XferBench is to be gauge of an EL's practical use in machine learning, it should seek to use a typologically diverse set of human languages in the downstream tasks. Second, since XferBench is trying to capture a general notion of "similarity to human language", it important to test this against a wide range of language typologies so as not to mistakenly narrow the criteria for "similar to human language". D3c is particularly important for incorporating interdisciplinary researchers into the field of EC who might not have a background in computer programming. Finally, D3d ensures that XferBench is accessible not only to labs and researchers with fewer financial resources but also makes it much easier to incorporate into the fast-paced research and development cycles prevalent in contemporary ML reserach.

[1] In the minority case, there are EC methods which use communication channels that are, for example, continuous (Eloff et al., 2021) or even pictorial (Mihai and Hare, 2021).

3.2 Methods

The following procedure describes the benchmark (illustrated in Figure 1):

1. Initialize a causal language model.
2. Train the model on the corpus of utterances from the EL being evaluated.
3. Re-initialize the input and output (i.e., language modelling head) embedding layers; this is the "base model".
4. For each downstream human language:
   (a) Train the base model on the human language data.
   (b) Evaluate the cross-entropy on a held-out test set of the human language.
5. Average the cross-entropies across the downstream human languages; this is the corpus's score on the benchmark (lower is better).

The structure of the benchmark is derived from the *corpus transfer* method presented in Yao et al. (2022).

Task For XferBench's evaluation task, we choose causal language modeling for a few different reasons. In principle, language modeling is a component of a wide variety of NLP tasks, especially generative tasks; the prevalence of language modeling is in line with the benchmark providing a very general notion of quality that will be familiar to anyone acquainted with NLP. On a practical level, language modeling is easy to acquire data for—especially helpful for evaluating against low-resource languages—and there are fewer hyperparameters and confounding variables compared to other downstream tasks like machine translation or question-answering. The main limitation from using language modeling is that it itself is not a widespread downstream task and so cannot guarantee direct correlation with metrics on more concrete downstream tasks (e.g., accuracy on a QA task).

For the pretraining task we also use causal language modeling. Due to requiring a wide applicability across emergent languages (Design Goal 2), we select causal language modeling for our pretraining task since it requires only a corpus without any additional annotations or stipulations.

Data The data for the transfer learning targets (viz. human languages) comes from Wikipedia dumps (Foundation) (under the GFDL and CC-BY-SA 3.0 License) hosted by Hugging Face[2]. This dataset provides a

[2] https://huggingface.co/datasets/wikimedia/wikipedia/tree/97323c5edeffcf4bd6786b4ed0788c84abd24b03

diverse set of languages each with sufficient amounts of data. For our downstream human languages, we use the same 10 languages presented in Yao et al. (2022), namely: Basque, Danish, Finnish, Hebrew, Indonesian, Japanese, Kazakh, Persian, Romanian, and Urdu. Having a variety of languages reduces the likelihood that XferBench will be biased toward specific typologies of human language (Design Goal 3b).

We use 15 and 2 million (10^6) tokens for the pretraining and fine tuning phases, respectively following Yao et al. (2022). Datasets are always repeated or truncated to fit the required size so that the number of training steps stays constant.

Tokenization For tokenization we use byte pair encoding (BPE) (Gage, 1994) with a vocabulary size of 30 000 for all human languages. Using BPE across all human languages is done primarily to simplify the implementation and keep tokenization methods consistent across all of the selected human languages. Emergent languages are generally considered to be pre-tokenized since most communication channels consist of one-hot vectors; thus, no additional tokenization or preprocessing is applied.[3]

Model For our model, we use a small configuration of GPT-2 (Radford et al., 2019), similar to that used in Yao et al. (2022): 6 attention heads, 6 layers, context length of 256, and hidden size of 768 with the remainder of the model parameters being the same as the defaults in the Hugging Face Transformers implementation.[4] This yields 65 million parameters in total. We kept the model on the smaller size to better suit it for the generally small amounts of data emergent languages corpora provide as well as to be more accessible (Design Goal 3d). Further details are listed in Appendix A.1.

Metric Given the use of language modeling for our evaluation task, we use token-level cross-entropy as the evaluation metric on the downstream task. This is a very common metric, making the outputs easy to interpret (Design Goal 3a). Although perplexity is more common as an evaluation of language models, the exponential nature of perplexity leads to more circuitous analyses and interpretation in our case, whereas cross-entropy is comparatively linear and additive (loosely speaking).[5] For the final score of the benchmark, we take the arithmetic mean of the cross-entropy across the 10 downstream human languages. That is, we define the benchmark's score for a given source language s as as h_s:

$$h_s = \underset{t \in T}{\text{mean}} \left(h_{s,t} \right) \tag{1}$$

where $h_{s,t}$ is the test cross-entropy of a model trained on source language s and finetuned and tested on target language t; T is the set of target languages. Since the score is based on cross-entropy, a lower score means better performance.

3.3 Implementation

XferBench is implemented as a small Python codebase which relies primarily on Hugging Face Transformers (Wolf et al., 2019) (Apache-2.0 license) and PyTorch (Paszke et al., 2019) (BSD-3-Clause license) libraries. To run the benchmark, all that is required is to install the environment with either pip or conda, and run `python -m xferbench path/to/corpus.jsonl` (Design Goal 3c). The input corpus is simply formatted as a newline-separated list of integer arrays, specifically in the JSON Lines format (see Appendix B for an example); a Hugging Face dataset (backed by Apache Arrow) can also be used for larger input corpora. The script executes all of the steps of the benchmark and yields a single floating point number which is that corpus's score on XferBench. Finer-grained functionalities are available and documented in the codebase. The benchmark takes about 5.5 hours to run on an NVIDIA GeForce RTX 2080 Ti: 90 minutes to train the base model and 30 minutes for tuning and testing on each of the target languages (Design Goal 3d).

The implementation is available at `https://example.com/benchmark-repo` under the MIT license (not published during review process; see supplementary materials for code).

4 Experiments

4.1 Procedures

XferBench The causal language modeling experiment is simply running XferBench as described in Section 3.2 on the reference and emergent languages discussed in Sections 4.2 and 4.3.

Machine translation The machine translation experiment is structured similarly to XferBench except with the downstream task being English-to-French translation (using the WMT 2014 dataset (Bojar et al., 2014)). The primary purpose of this experiment is to determine how well XferBench correlates with a more concrete downstream task (especially one that incorporates language modeling). We choose this language pair in part to gauge the relative differences between the task languages and the baseline human languages (in contrast to XferBench which we want to be largely agnostic to human languages). Looking at our reference human languages, we have: French, the target language itself; Spanish, closely related to French; Russian and Hindi, distantly related to French; and Chinese, Korean

[3] Whether the tokens of an EL should be treated as words or subword units is an open question, although tokens as words is more common (but see Ueda et al. (2023) for tokens as subword units). Practically speaking, many emergent languages are small enough that applying a 30 000-item BPE model would severely reduce the corpus size.

[4] https://huggingface.co/docs/transformers/v4.36.1/en/model_doc/gpt2#transformers.GPT2Config

[5] For example, it would make more sense to use log scales and geometric means to average and compare perplexities, but this would just be reverting back to cross-entropy!

and Arabic, not related to French. Instead of using a GPT-2–based model, we use a BART-based model since MT is a conditional generation task (see Appendix A.2 for details). The pretraining dataset size is increased to 100 million due to the increased difficulty of this task compared to language modeling. We evaluate the translation performance with chrF (Popović, 2015) and BLEU (Papineni et al., 2002) using the default Hugging Face Evaluate metrics (derived from sacreBLEU (Post, 2018)). Evaluation is performed with beam sizes of 1, 3, and 5, and the resulting values are averaged.

We present three settings for this experiment. The first is *Full* which tunes on 50 million source tokens at a higher learning rate ($1 \cdot 10^{-4}$ for training and $2 \cdot 10^{-4}$ for the AdamW optimizer (Kingma and Ba, 2015)), which we found empirically to lead to the best performance. The second is *Frozen*, in which we use the same configuration as *Full* but freeze all but the embedding layers before tuning the model for translation (as in Papadimitriou and Jurafsky (2020); Artetxe et al. (2020)). Finally, we also present *Reduced* which uses a smaller tuning dataset of 10 million tokens and lower learning learning ($2 \cdot 10^{-5}$); the lower rate helped the random baselines converge better as well as showed better distinction between languages.

4.2 Reference languages

The following reference languages serve as a way to contextualize the results of XferBench as well as to validate that it is capturing some notion of the quality of the emergent languages (cf. Section 4.4).

Human languages For our baseline line human languages, we selected French, Spanish, Russian, Chinese, Korean, Arabic, and Hindi.[6] Like the evaluation languages, the data is derived from Wikipedia articles (same source as the target languages).

Synthetic languages For synthetic languages, we follow Yao et al. (2022) and use "Zipfian parentheses" from Papadimitriou and Jurafsky (2020). This synthetic dataset—referred to as *Paren, real*—is hierarchically balanced "parentheses" where each parenthesis is the token ID sampled from the unigram distribution of a human language (hence "Zipfian"). This datasets mimics both the unigram distribution of a human language as well as the basic recursive hierarchical structure. This yields a reasonably strong yet simple baseline for synthetic data.

We also test a fully synthetic dataset (*Paren, synth*) which uses the same hierarchical parenthesis generation script from Papadimitriou and Jurafsky (2020), replacing the data-derived unigram distribution with Zipf–Mandelbrot distribution:

$$f(w_i) = \frac{1}{(i+\beta)^\alpha} \quad (2)$$

[6]The main reason for choosing the high-resource language is due to the higher data requirements of machine translation experiment discussed below.

Setting	Observ.	$\lvert V\rvert$	$\lvert M\rvert$	$\lvert C\rvert$
Disc, small	one-hot	6	11	700
Disc, large	one-hot	100	31	100 M
Recon, large	one-hot	100	31	31 M
Mu+, CUB	embed	20	10	1.3 M
Mu+, SW	embed	14	7	1.2 M
Yao+	embed	4028	15	43 M

Table 1: Summary of key hyperparameters in the tested emergent languages. Observations are either one-hot vectors or embeddings. $|V|$, $|M|$, and $|C|$ refer to the vocabulary, message, and corpus size respectively.

where $f(w_i)$ is non-normalized probability weight of word w with 1-based index (rank) i, $\alpha = 1$, $\beta = 2.7$ (Mandelbrot et al., 1953; Piantadosi, 2014).

Random baselines We use two random baselines. The first is simply a uniform unigram distribution across the whole vocabulary with no additional structure (referred to as *Random*). This baseline sheds light on whether the optimization itself, no matter training data, primes the network in some way for transfer learning. The second "random" baseline is no pretraining at all (*No pretrain*); that is, a network which has been freshly initialized at the tuning stage. This baseline helps establish whether the pretraining on other languages has any impact beyond the tuning data in isolation.

4.3 Emergent languages

We present a summary of the key hyperparameters of emergent languages in Table 1. The emergent language corpora below come from reproductions from existing codebases with the exception of Yao et al. (2022), whose emergent language corpus is available for download. Emergent languages which have a corpus size smaller than the required size are simply repeated and shuffled as many times as necessary so that the model receives the same number of optimization steps.

Generic signalling game The first set of emergent languages we test are generic versions of the of the signalling game (reference game) as implemented in EGG (Kharitonov et al., 2019) (MIT license). These games use one-hot vectors to represent attribute–value observations, that is, observations are elements of the set $V^{|A|}$ where V is the set of values and $|A|$ is the number of attributes. The signalling game is one of the simplest and most used games in emergent communication research.

The first two language are *Disc, small* and *Disc, large* which are two configurations of the discrimination version of the signalling game. Here, the sender makes an observation and sends a message; then, the receiver must select the corresponding observation from a small set of potential observations (like a multiple-choice question). The *small* configuration consists of 4 attributes and 4 values with a small vocabulary size and medium message length; this setting is intended to represent a toy

environment that one might find in an emergent communication paper. The *large* configuration consists of 12 attributes and 8 values with a larger vocabulary and longer message length. Both environments show 5 distractor observations to the receiver (i.e., 6-way multiple choice). Both settings converge to a success rate $>95\%$ compared to a random baseline of 17%.

The *Recon, large* environment is based on the reconstruction version of the signalling game. In this version, the receiver does not make any observations and instead must recreate the sender's observation based on the message alone (similar to an autoencoder). The observation space has 8 attributes and 8 values with other settings identical to that of *Disc, large*. Since the reconstruction game considerably harder, the game does not converge but does reach an overall accuracy of 0.014% and per-attribute accuracy of 24% compared to a random baseline of 0.0000060% and 13% random baseline, respectively. For details, see Appendix A.3.

Mu and Goodman (2021) present the second pair of emergent languages which we test XferBench on (code under MIT license). The emergent communication game is also a discriminative signalling game but with (1) richer observations and (2) more abstract information needing to be communicated. In one setting, the observations are images from ShapeWorld (Kuhnle and Copestake, 2017) (*Mu+, SW*), a synthetic data of various geometric shapes, and the other setting is CUB (Wah et al., 2011) (*Mu+, CUB*) which contains labeled images of birds; both settings encode features with a CNN which is the passed to the sender and receiver. In the basic discriminative game, the observation made by the sender is the exact same one seen by the receiver. Mu and Goodman (2021) instead uses a "concept game" where the sender must communicate some abstract concept shared by a set of input images which the receiver will then have to a pick out from a different set of images, some sharing the same concept (e.g., isolating the concept of "triangle" or "bird size"). The ShapeWorld and CUB games had test accuracies of 71% and 66% respectively compared to a random baseline of 50%, comparable to the reported values in the paper. All messages were taken from observations seen in training.

Yao et al. (2022) present a standard discrimination game which uses natural images (Conceptual Captions (Sharma et al., 2018) (images only)) as inputs to the sender and receiver (code unlicensed but distributed on GitHub with paper). The accuracy for the particular emergent language corpus is not reported in the paper, but comparable experiments from the paper would suggest that it converged to an accuracy of $>90\%$ compared to a baseline of 0.4% (i.e., 255 distractors).

4.4 Hypotheses

The following hypotheses are directly relate to determining whether or not XferBench is quantifying some meaningful notion of the quality of a language (i.e., Design Goal 1).

(H1) Human languages will perform best, followed by the synthetic and emergent languages, followed by the random baselines.

(H2) Human languages will have similar performance on XferBench (also key for Design Goal 3b); the intuition here is that human languages share deep structural similarities. This hypothesis is supported, in part, by Artetxe et al. (2020). For the MT experiment, we expect to see the following order of performance based on language relatedness: {*French*}, {*Spanish*}, {*Russian*, *Hindi*}, {*Chinese*, *Korean*, *Arabic*}.

(H3) Languages with a larger vocabulary, longer message length, and larger corpora will perform better. In particular, we expect *Disc, large* will perform better than *Disc, small* since the former is a more "complex" version of the latter. This hypothesis (for vocabulary size and message length) is supported by some experiments in Yao et al. (2022, app. B.4).

(H4) XferBench will correlate well with scores on the machine translation task (i.e., cross-entropy will correlate negatively with chrF).

5 Results

5.1 XferBench

In Figure 2 we show the results of the benchmark (i.e., causal language modeling) on the various baselines. Each mean is displayed with error bars showing the 95% confidence interval of mean as calculated with bootstrapping (details in Appendix E). For reference, the cross-entropies range from about 5.2 to 5.5 corresponding to perplexities of 180 to 240.

The human languages show the best score (lowest cross-entropy) on the benchmark with *Chinese*, *Korean*, and *Arabic* performing the best in one cluster and *French*, *Spanish*, *Russian*, and *Hindi* performing slightly worse in their own cluster (based on confidence intervals). The synthetic and emergent languages all show similar performance with only small variations with the exception of the *Disc, large* language which is better than the rest of the emergent languages but still worse than the human languages. Finally, the random baselines perform worse than the rest of the tested languages. *No pretrain*'s performance is worse than the cluster of synthetic and emergent languages but better than the fully random language (*Random*).

5.2 Machine Translation

The chrF scores of the machine translation experiment are given in Table 2 (BLEU scores in Appendix D.1). Additionally, we give Pearson correlation coefficients between each setting and the scores generated by XferBench (scatter plots shown in Appendix D.3). In all settings, we see that XferBench is strongly correlated with the results of the machine translation experiment.

For the *Full* setting, the results are somewhat inconclusive. Human languages perform the best and similarly to each other. *Paren, real*, *Paren, syn*, *Disc, large*, and *Mu+, CUB* all match the performance of human

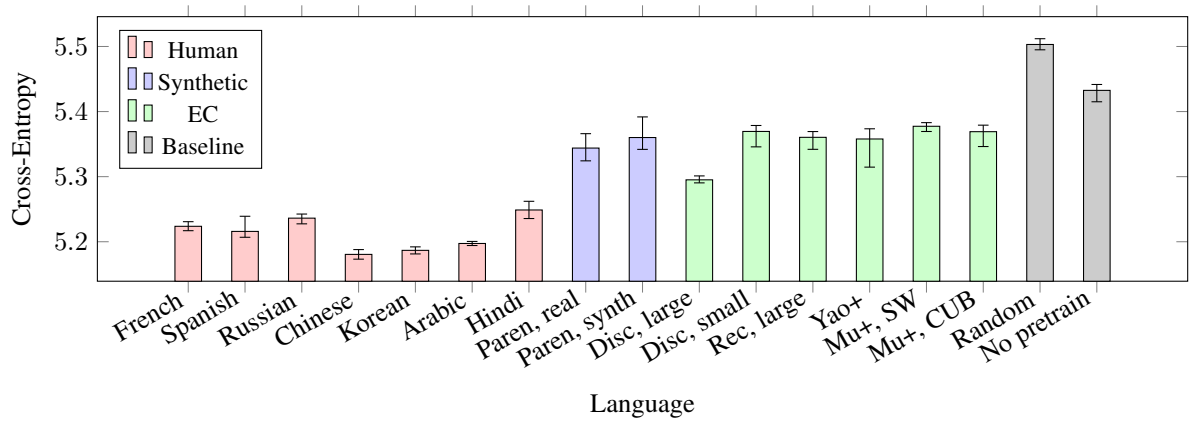

Figure 2: Average cross-entropy on target language datasets for each source language. Lower is better. Error bars represent 95% confidence intervals.

Source	Full	Frozen	Reduced
French	47.8	31.4	35.8
Spanish	48.0	27.9	34.8
Russian	47.6	29.0	37.2
Chinese	47.5	22.2	35.2
Korean	47.7	23.3	35.6
Arabic	47.8	27.6	36.6
Hindi	47.5	26.0	31.7
Paren, real	47.5	10.5	35.0
Paren, synth	48.2	12.0	34.3
Disc, large	47.7	24.7	30.7
Disc, small	14.3	16.2	17.3
Rec, large	22.5	18.4	25.4
Yao+	4.0	20.1	25.6
Mu+, SW	3.3	18.4	23.3
Mu+, CUB	47.6	21.6	24.6
Random	1.8	3.0	19.7
No pretrain	11.4	4.3	28.1
Correl. with XferBench	−0.75	−0.84	−0.79

Table 2: chrF scores across three English-to-French machine translation settings. Correlation measured with the Pearson correlation coefficient.

languages as well. The rest of the language perform significantly worse than the aforementioned languages, especially *Yao+* and *Mu+, SW* (see Appendix F for sample outputs). In the case of *Random*, the training loss did not decrease during training likely due to the high learning rate.

In *Frozen*, we see the best correlation with the hypothesis regarding human languages (as well as with XferBench itself). *Disc, large* performs comparably to the worst human languages and better than the rest of the languages. The remainder of the synthetic and emergent languages perform worse than the human languages but better than the random baselines.

Finally, *Reduced* (i.e., lower learning rate and tuning data) displays better separation than *Full*, but not as significant as *Frozen*. Human languages still perform the best, although they are matched by the *Paren* languages. *Disc, large* underperforms the human languages but still outperforms all other emergent languages. All emergent languages, apart from *Disc., large* underperform the *No pretrain* baseline. The better half of languages performed better (compared to themselves) with a higher learning rate while the lower half performed better with a reduced learning rate.

6 Discussion

6.1 Experiments

The basic ordering of the language by XferBench follows basic *a priori* assumptions: random baselines pretrain the worst, human languages perform the best, and emergent and synthetic languages are bounded above and below by these (supporting Hypothesis 1). Human languages cluster together in XferBench although there is still variation with non-overlapping confidence intervals (partially supporting Hypothesis 2).

Intra-EL differences Generally speaking, there is very little variation shown by XferBench on the emergent languages; nevertheless, we can still draw a handful of conclusions. First, *Disc, large* outperforms *Disc, small* while sharing the same codebase, task, etc. and differing only in message length, vocabulary size, observation space, and corpus size (supporting Hypothesis 3). This result matches the trend seen in Yao et al. (2022) that larger vocabularies and message lengths in an emergent language lead to better performance on downstream data. On the other hand, *Disc, small* performs similarly to other languages with larger vocabularies and longer message lengths (contradicting Hypothesis 3).

Second, it seems that the underlying complexity of the emergent communication game does not directly correlate with XferBench score: the abstract visual reasoning of *Mu+, SW* and *Mu+, CUB* does not lead to it outperform *Disc, small*. Additionally, the richer observations (i.e., image embeddings) of *Mu+, CUB* and *Yao+*

also do not, by their mere presence, confer an advantage to the emergent language with respect to XferBench.

Finally, *Disc, large* and *Recon, large* both share hyperparameters in terms of the vocabulary size, message length, and corpus size, yet *Disc, large* shows significantly better performance on XferBench. This indicates that XferBench is not *solely* concerned with surface-level features as the nature of the game (e.g., discrimination versus reconstruction, success rate) is relevant as well.

Correlation with MT The results from the machine translation experiment show strong, though not perfect, (negative) correlation with XferBench (supporting Hypothesis 4). For example, in all cases, *Disc, large* outperforms all other emergent languages. This strongly supports the notion that XferBench performance is predictive of downstream performance on more concrete NLP tasks.

The results from the *Full* setting of the MT experiment do show some correlation with XferBench but fail to show expected trends in other ways. For example, there is no clear ordering among the human languages (e.g., *French* does *not* outperform *Arabic*). Additionally *Yao+* and *Mu+, SW* drastically underperform the other emergent languages and the *No pretrain* baseline. We suspect that these aberrations from expected results come in part due to the high learning rate which cause unstable training or generation. On the other hand, the *Frozen* setting gives us the clearest ordering of human languages that matches with *a priori* expectations; this setting also has the strongest correlation with XferBench scores. The *Reduced* setting shows better correlation than *Full* but is not as clear as *Frozen*.

Random baselines In all of our experiments, the pretraining on random tokens (*Random*) performed notably worse than not pretraining at all (*No pretrain*), suggesting that ill-conditioning the neural network can be a significant hindrance to performing well on XferBench. This is important to note in light of the fact that a perfectly one-to-one compositional language describing uniformly sampled attribute–value vectors would yield a corpus with a uniformly random unigram distribution. This is to say, a fully compositional language, which is often seen as desirable in emergent communication research, could make for a very poor source of pretraining data as shown by *Random*'s performance on XferBench.

This fact along with the observations about sensitivity to learning rate indicates that performance on XferBench is not simply a function of the particular features of the emergent language in relation to the downstream human languages but also a function of the dynamics of optimization (i.e., priming the model to adapt well). Although this increases the difficulty of developing and interpreting a tool like XferBench, it is almost an unavoidable part of deep learning methods.

6.2 Future work

We identify three main directions for future work with XferBench. The first direction is determining what XferBench is measuring and how its scores correlate with the different factors of emergent languages. Yao et al. (2022, app. B.4) pursued this on a small scale with factors like vocabulary size and message length, but there exist a host of other factors worth exploring: speaker model size, game design, language entropy, observation modality, etc.

The second direction is more extensively investigating the correlation of XferBench with downstream tasks. We would expect that tasks that rely heavily on a language model—such as automatic speech recognition, abstractive summarization, and generative question-answering—to correlate well with XferBench. On the other hand, tasks that are more focused on classification—such as named entity recognition, sentiment analysis, and multiple choice question-answering—might not correlate as well.

Finally, XferBench would benefit greatly from improved compute efficiency. For example, if the results of XferBench could be replicated with a fraction of the training steps, it could (1) allow for a larger number of downstream languages to be tested which would reduce the size of the confidence intervals, allowing more more precise scoring. And (2), it would open the door to using larger models which would better capture the deeper structures of language and likely correlate better with realistic downstream tasks.

7 Conclusion

In this paper we have introduced XferBench, a first-of-its-kind benchmark for evaluating the quality of an emergent language corpus based on its transfer learning performance on human languages. This approach to evaluating emergent language scales with data and compute as opposed to requiring increasingly complex handcrafted rules to measure the desirable qualities of emergent language. We provide empirical results of XferBench across human, synthetic, and emergent languages and demonstrate that these results correlate with downstream performance on a machine translation task. XferBench is implemented as an easy-to-use Python package that will permit researchers in the field to easily apply XferBench to new emergent languages.

8 Limitations

The first limitation of XferBench is that it relies on a restricted interface with the emergent communication system. With emergent communication we have access not only to the grounding of all of the utterances of the emergent language but also full access to the agents themselves. Language is fundamentally a contextual phenomenon, so only a small part of it can be understood from looking at corpora in isolation. Thus, although XferBench is much more broadly applicable

because of this restricted interface, it is also quite limited in what it can detect from a theoretical point of view.

The other set of limitations we will discuss have to do with the model and data size. First, the model and data size (60 M parameters and 15 M tokens) are quite small by contemporary standards, limiting the direct applicability of results from XferBench to relevant downstream tasks involving large language models, for example. On the other hand, scaling up the models, data, and methods of XferBench comes with its own difficulties. First, it would start to bias the benchmark towards high-resource languages, as only those could provide the necessary data to accommodate larger models. Second, it would make XferBench, which is already relatively slow as a metric (6 GPU-hours) even slower. This would decrease the speed of the iterative design process of emergent communication systems and, thus, the utility of the metric as a whole.

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.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Ale s Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Henry Brighton and Simon Kirby. 2006. Understanding linguistic evolution by visualizing the emergence of topographic mappings. *Artificial Life*, 12:229–242.

Kalesha Bullard, Franziska Meier, Douwe Kiela, Joelle Pineau, and Jakob N. Foerster. 2020. Exploring zero-shot emergent communication in embodied multi-agent populations. *ArXiv*, abs/2010.15896.

C.m. Downey, Xuhui Zhou, Zeyu Liu, and Shane Steinert-Threlkeld. 2023. Learning to translate by learning to communicate. In *Proceedings of the 3rd Workshop on Multi-lingual Representation Learning (MRL)*, pages 218–238, Singapore. Association for Computational Linguistics.

Kevin Eloff, Arnu Pretorius, Okko Johannes Räsänen, Herman Arnold Engelbrecht, and Herman Kamper. 2021. Towards learning to speak and hear through multi-agent communication over a continuous acoustic channel. *ArXiv*, abs/2111.02827.

Wikimedia Foundation. Wikimedia downloads.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38.

Lukas Galke, Yoav Ram, and Limor Raviv. 2022. Emergent communication for understanding human language evolution: What's missing? *ArXiv*, abs/2204.10590.

Yuxuan Guo, Yifan Hao, Rui Zhang, Enshuai Zhou, Zidong Du, Xishan Zhang, Xinkai Song, Yuanbo Wen, Yongwei Zhao, Xuehai Zhou, Jiaming Guo, Qi Yi, Shaohui Peng, Di Huang, Ruizhi Chen, Qi Guo, and Yunji Chen. 2023. Emergent communication for rules reasoning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Eugene Kharitonov, Rahma Chaabouni, Diane Bouchacourt, and Marco Baroni. 2019. EGG: a toolkit for research on emergence of lanGuage in games. 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): System Demonstrations*, pages 55–60, Hong Kong, China. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Alexander Kuhnle and Ann A. Copestake. 2017. Shapeworld - a new test methodology for multimodal language understanding. *ArXiv*, abs/1704.04517.

Travis LaCroix. 2019. Biology and compositionality: Empirical considerations for emergent-communication protocols. *CoRR*, abs/1911.11668.

Angeliki Lazaridou and Marco Baroni. 2020. Emergent multi-agent communication in the deep learning era. *ArXiv*, abs/2006.02419.

Angeliki Lazaridou, Karl Moritz Hermann, Karl Tuyls, and Stephen Clark. 2018. Emergence of linguistic communication from referential games with symbolic and pixel input. *ArXiv*, abs/1804.03984.

Fushan Li and Michael Bowling. 2019. *Ease-of-Teaching and Language Structure from Emergent Communication*. Curran Associates Inc., Red Hook, NY, USA.

Yaoyiran Li, Edoardo Maria Ponti, Ivan Vulić, and Anna Korhonen. 2020. Emergent communication pretraining for few-shot machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4716–4731, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Benoit Mandelbrot et al. 1953. An informational theory of the statistical structure of language. *Communication theory*, 84:486–502.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: The penn treebank. *Comput. Linguistics*, 19:313–330.

Daniela Mihai and Jonathon S. Hare. 2021. Learning to draw: Emergent communication through sketching. In *Neural Information Processing Systems*.

Clément Moulin-Frier and Pierre-Yves Oudeyer. 2020. Multi-agent reinforcement learning as a computational tool for language evolution research: Historical context and future challenges. *ArXiv*, abs/2002.08878.

Jesse Mu and Noah Goodman. 2021. Emergent communication of generalizations. In *Advances in Neural Information Processing Systems*, volume 34, pages 17994–18007. Curran Associates, Inc.

Yao Mu, Shunyu Yao, Mingyu Ding, Ping Luo, and Chuang Gan. 2023. Ec2: Emergent communication for embodied control. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6704–6714.

Isabel Papadimitriou and Dan Jurafsky. 2020. Learning music helps you read: Using transfer to study linguistic structure in language models. In *Conference on Empirical Methods in Natural Language Processing*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Hugh Perkins. 2022. Icy: A benchmark for measuring compositional inductive bias of emergent communication models.

Steven T Piantadosi. 2014. Zipf's word frequency law in natural language: a critical review and future directions. *Psychon. Bull. Rev.*, 21(5):1112–1130.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia. Association for Computational Linguistics.

Ryo Ueda, Taiga Ishii, and Yusuke Miyao. 2023. On the word boundaries of emergent languages based on harris's articulation scheme. In *The Eleventh International Conference on Learning Representations*.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. 2011. The caltech-UCSD birds-200-2011 dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.

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. In *BlackboxNLP@EMNLP*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Shunyu Yao, Mo Yu, Yang Zhang, Karthik R Narasimhan, Joshua B. Tenenbaum, and Chuang Gan. 2022. Linking emergent and natural languages via corpus transfer. In *International Conference on Learning Representations*.

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.

A Hyperparameters

A.1 Causal language modeling

For values not listed, see Hugging Face Transformers' defaults at `https://huggingface.co/docs/`

transformers/v4.36.1/en//model_doc/gpt2#transformers.GPT2Config.

- Model: GPT-2
- Tokenizer: Byte pair encoding
- Hidden size: 768 (default)
- Vocabulary size: 30 000
- Context length: 256
- Number of layers: 6
- Number of attention heads: 6
- Learning rate: $1 \cdot 10^{-4}$
- Optimizer: AdamW
- Weight decay: 0.01
- Learning rate schedule: linear (to 0)
- Batch size: 32
- Train dataset size: $15 \cdot 10^6$ tokens
- Train epochs: 5
- Tune dataset size: $2 \cdot 10^6$ tokens
- Train epochs: 10

A.2 Machine translation

For values not listed, see Hugging Face Transformers' defaults at https://huggingface.co/docs/transformers/v4.36.1/en/model_doc/bart#transformers.BartConfig. The following is for the *Full* setting.

- Model: BART
- Training objective: text infilling only (see note below)
- Tokenizer: Byte pair encoding
- Hidden size: 512
- Vocabulary size: 30 000
- Context length: 512
- Number of encoder layers: 6
- Number of decoder layers: 6
- Number of encoder attention heads: 8
- Number of decoder attention heads: 8
- Encoder feedforward dimension: 2048
- Decoder feedforward dimension: 2048
- Train learning rate: $1 \cdot 10^{-4}$
- Tune learning rate: $2 \cdot 10^{-4}$
- Optimizer: AdamW
- Weight decay: 0.01
- Learning rate schedule: linear (to 0)
- Batch size: 32
- Train dataset size: $100 \cdot 10^6$ tokens
- Train epochs: 5
- Tune dataset size: $50 \cdot 10^6$ tokens
- Train epochs: 3
- Test beam size: $1, 3, 5$ (final metric averaged across each size)
- Test context size: 128

The objective used to pretrain BART was text infilling *only*; we cannot use the sentence permutation objective because we do not know *a priori* what constitutes a sentence in an emergent language, hence we do not use it for any settings. For the *Frozen* setting, all is as above, but all non-embedding layers are frozen for the duration of tuning. For the *Reduced* setting, all is as above except for the following:

- Tune learning rate: $1 \cdot 10^{-5}$
- Tune dataset size: $10 \cdot 10^6$

A.3 Generic signalling game

We use the following hyperparameters for the *Disc, small* emergent language.

- Game (from EGG): `egg.zoo.basic_games.play`
- Message optimization: Gumbel-softmax (as opposed to REINFORCE)
- Game type: discrimination
- Number of attributes: 4
- Number of values: 4
- Number of distractors: 5
- Vocabulary size: 6
- Max message length: 10
- Number of examples: 32 768
- Batch size; 1024
- Number of epochs: 10
- Sender hidden size: 256
- Receiver hidden size: 512
- Sender embedding size: 32
- Receiver embedding size: 32
- Sender network type: GRU
- Receiver network type: GRU
- Learning rate: 0.001

The *Disc, large* setting uses the same hyperparameters as above with the exception of the following.

- Number of attributes: 12
- Number of values: 8
- Number of distractors: 5
- Number of examples: $3.5 \cdot 10^6$
- Max message length: 30
- Vocabulary size: 100
- Number of epochs: 100

The *Recon, large* setting is as in *Disc, large* with the following changes.

- Game type: reconstruction
- Number of attributes: 8
- Number of distractors: N/A
- Number of examples: $1 \cdot 10^6$
- Number of epochs: 10

B Example of benchmark input format

The input format for the benchmark is simple: integer arrays in a JSON format separated by newlines (i.e., JSON Lines, JSONL, `*.jsonl`). The following is an example of file contents in this format:

```
[3, 1, 4, 1, 5, 9, 2]
[6, 5, 3, 5, 8, 9, 7, 9, 3]
[2, 3, 8, 4]
[6, 2, 6, 4, 3, 3]
[8, 3, 2, 7, 9, 5, 0, 2, 8, 8, 4]
```

C Computing resources used

See Table 3 for rough estimates of the compute used in writing this paper. Most experiments were run on a shared cluster comprising approximately 150 NVIDIA A6000 (or comparable) GPUs.

Item	Base GH	n items	Total
XferBench	6	45	270
MT	8	50	400
Other experiments	2	50	100
Total			770

Table 3: Estimate of compute used for this paper in GPU-hours (specifically NVIDIA RTX 2080 Ti–hours).

Source	Full	Frozen	Reduced
French	12.93	5.33	6.61
Spanish	13.32	4.52	6.35
Russian	12.93	4.37	7.02
Chinese	12.71	3.04	6.03
Korean	12.83	2.95	6.36
Arabic	13.12	4.16	6.74
Hindi	12.72	3.20	5.24
Paren, real	12.60	0.65	6.26
Paren, synth	13.19	0.82	6.15
Disc, large	12.93	2.08	4.44
Disc, small	0.17	0.19	0.38
Rec, large	1.92	0.86	2.50
Yao+	0.01	1.04	2.57
Mu+, SW	0.00	1.05	1.86
Mu+, CUB	12.71	1.45	2.35
Random	0.00	0.00	1.02
No pretrain	0.10	0.06	3.43

Table 4: BLEU scores for machine translation experiment.

D Additional results

D.1 BLEU scores for machine translation

See Table 4.

D.2 Raw cross-entropies on XferBench

See Table 5.

D.3 Scatter plots for XferBench and MT

See Figure 3.

E Cross-entropy confidence interval computation

Let $s \in S$ and $t \in T$ represent source and target languages, respectively. $h_{s,t}$ represents the test cross-entropy of a model pretrained on s and evaluated on t. As sated in Equation (1), the score on XferBench is the mean cross-entropy across all target languages:

$$h_s = \operatorname*{mean}_{t' \in T} \left(h_{s,t'} \right). \tag{3}$$

We would like to calculate a confidence interval (i.e., h_s^- and h_s^+) for a source language's mean cross-entropy using the different cross-entropies on the target languages (i.e., $h_{s,t}$ for $t \in T$), yet these samples are not i.i.d., since the mean of cross-entropy each *target* language can vary. Thus, if we would like to use bootstrapping to calculate confidence intervals, we must first normalize the cross-entropies. Let $\hat{h}_{s,t}$ be the normalized score:

$$\hat{h}_{s,t} = \frac{h_{s,t} - \text{mean}_{s' \in S} \left(h_{s',t} \right)}{\text{stdev}_{s' \in S} \left(h_{s',t} \right)}. \tag{4}$$

Given the normalized scores, we can now bootstrap in order to compute confidence intervals for $\hat{h}_s$ (i.e., in the normalized space).[7] Let $\hat{h}_s^+$ and $\hat{h}_s^-$ be the upper and lower bounds of the confidence interval computed using bootstrapping in the normalized space. We can now translate these back into the raw cross-entropy space using the means and standard deviations from before:

$$h_s^+ = \hat{h}_s^+ \cdot \operatorname*{stdev}_{s' \in S} \left(h_{s',t} \right) + \operatorname*{mean}_{s' \in S} \left(h_{s',t} \right) \tag{5}$$

$$h_s^- = \hat{h}_s^- \cdot \operatorname*{stdev}_{s' \in S} \left(h_{s',t} \right) + \operatorname*{mean}_{s' \in S} \left(h_{s',t} \right). \tag{6}$$

F Error analysis

In the *Full* setting of the machine translation task, the *Yao+* and *Mu+, SW* settings perform worse than expected (*a priori* and compared to the other results in the setting). Validation loss converged while chrF and BLEU scores remained near zero. We provide a couple examples (taken from the predefined test set of WMT 2014) of model output to provide some insight into the reason for this. No post processing used, generation is capped at 50 tokens, and "\u0000" represent single non-printable characters.

[7] This is not intended to be statistically rigorous. Our cross-entropies are unlikely to be normally distributed, but this still be helpful for generally gauging uncertainty.

Source	Danish	Basque	Persian	Finnish	Hebrew	Indonesian	Japanese	Kazakh	Romanian	Urdu
French	4.93	6.03	5.04	5.62	5.48	4.87	5.23	5.46	5.15	4.43
Spanish	4.92	6.06	5.03	5.61	5.47	4.82	5.25	5.46	5.12	4.42
Russian	4.94	6.04	5.04	5.65	5.48	4.88	5.27	5.48	5.14	4.45
Chinese	4.89	6.02	5.01	5.58	5.43	4.76	5.18	5.44	5.12	4.39
Korean	4.89	6.01	5.02	5.57	5.44	4.78	5.20	5.45	5.12	4.38
Arabic	4.90	6.02	5.02	5.59	5.45	4.81	5.22	5.44	5.13	4.40
Hindi	4.94	6.06	5.08	5.65	5.47	4.83	5.29	5.52	5.20	4.46
Paren, real	5.07	6.11	5.11	5.75	5.59	5.06	5.38	5.57	5.22	4.56
Paren, synth	5.08	6.13	5.14	5.74	5.58	5.09	5.43	5.58	5.26	4.57
Disc, large	5.00	6.06	5.11	5.71	5.52	4.92	5.34	5.56	5.25	4.49
Disc, small	5.09	6.06	5.17	5.80	5.59	5.05	5.41	5.65	5.31	4.56
Rec, large	5.09	6.06	5.16	5.79	5.57	5.04	5.41	5.64	5.30	4.55
Yao+	5.07	6.03	5.17	5.79	5.56	5.03	5.41	5.65	5.31	4.56
Mu+, SW	5.09	6.10	5.18	5.80	5.58	5.05	5.42	5.65	5.33	4.58
Mu+, CUB	5.08	6.06	5.18	5.79	5.58	5.05	5.42	5.65	5.32	4.56
Random	5.23	6.17	5.31	5.92	5.71	5.22	5.55	5.76	5.45	4.72
No pretrain	5.17	6.10	5.23	5.85	5.66	5.14	5.47	5.68	5.38	4.65

Table 5: Cross-entropies across all source and target languages.

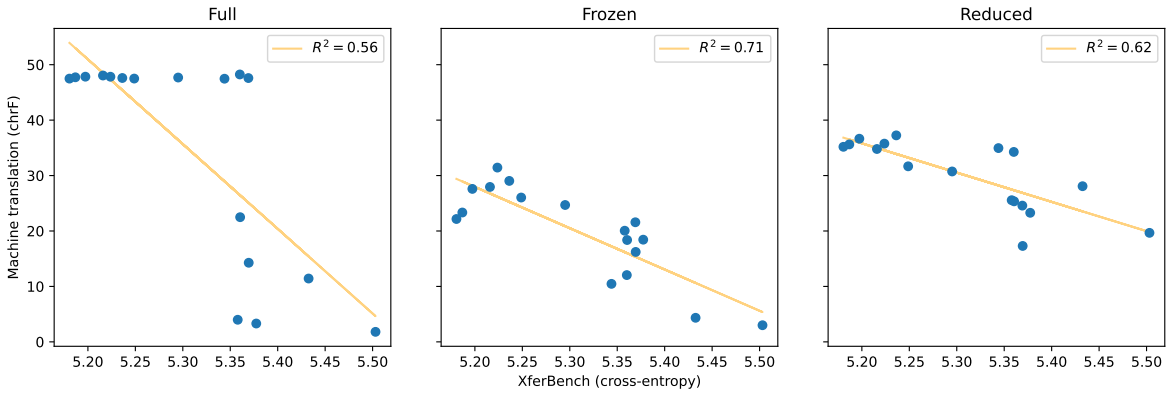

Figure 3: Scatter plots showing XferBench score versus machine translation score.

Example 1 *Input*: "And while Congress can't agree on whether to proceed, several states are not waiting."

Reference: "Et tandis que les membres du Congrès n'arrivent pas à se mettre d'accord pour savoir s'il faut continuer, plusieurs États n'ont pas attendu."

[Model pretrained on] *French*: "#Et alors que le Congrès ne peut pas convenir de poursuivre, plusieurs États ne sont pas en attente. » (traduction libre) Le Parlement européen. Le Parlement européen est d'avis que le Parlement européen doit être en mesure de faire preuve#"

Disc, large: "#Et bien que le Congrès ne puisse pas convenir de la marche à suivre, plusieurs États ne sont pas en attente.\u2028\u2028[Traduit par la Rédaction]\u2028(Traduit par la Rédaction)\u2028(Tra#"

Yao+: "#Annexe II, par.\xa02.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.#"

Mu+, SW: "#Annexe II.\xa0\xa0: Appendice I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I#"

Example 2 *Input*: "This really is a must for our nation."

Reference: "Cela est vraiment indispensable pour notre nation."

French: "#C'est vraiment une nécessité pour notre nation. Nous devons y parvenir. Nous devons y parvenir. Nous devons y parvenir. Nous devons y parvenir. Nous devons y parvenir. Nous devons y parvenir. Nous devons y#"

Disc, large: "#C'est vraiment un devoir pour notre nation. C'est un devoir.\u2028\u2028(...)\u2028\u2028(...)\u2028(...)\u2028(...)\u2028(...)\u2028(...)\u2028(...)\u2028(#"

Yao+: "#Annexe II, par.\xa02.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.2.#"

Mu+, SW: "#Annexe II.\xa0\xa0: Appendice I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I.\xa0I#"

Discussion Although all of the models have trouble terminating properly, the *French* and *Disc, large* models (which have high chrF scores) clearly condition their generation on the text, whereas *Yao+* and *Mu+, SW* give the same output regardless of the input. Although this is unexpected, we can see in the *Full* setting in Figure 3 that there is sharp drop off between high-performing and low-performing languages. We suspect that the higher learning rate during tuning caused this bimodal distribution of results and is at least in part responsible for the poor performance *Yao+* and *Mu+, SW* models on the MT experiment's *Full* setting.