Learning Phonotactics in a Differentiable Framework of Subregular Languages

Huteng Dai[1] & Richard Futrell[2]
[1]*Rutgers University,* [2]*University of California, Irvine*

1 Introduction

Phonotactic constraints have been argued to be **regular**, meaning that they can be represented using finite-state automata (Heinz, 2018); furthermore, they have been argued to occupy a even more restricted region of the regular language class known as the subregular hierarchy (Rogers & Pullum, 2011). Our contribution is to present a simple model of phonotactic learning from positive evidence. Our approach is based on probabilistic finite-state automata (Vidal et al., 2005a,b). We study the model's ability to induce local and nonlocal phonotactics from wordlist data, both with and without formal constraints on the automaton. In particular, we evaluate the ability of our learner to induce nonlocal phonotactic constraints from data of Navajo and Quechua. Our work provides a framework in which different formal models of phonotactics can be compared, and sheds light on the structural nature of phonological acquisition (Dai, 2021; Shibata & Heinz, 2019; Heinz & Rogers, 2010, 2013).

2 Background: Formal properties of phonotactics

2.1 *Phonotactics and formal grammar* Phonotactics is the speakers' knowledge of legal and illegal sound sequences in a language. In formal language theory, a **language** is defined as a set of strings (or **words**). For example, given a segmental inventory {s, o, ʃ}, two possible languages are A and B, as illustrated in Table 1. Those strings that belong to the set, such as {soʃ, sosoʃ} in language A, are legal, and those that do not belong to the set, such as {*sʃo, *sosʃ, . . . }, are illegal.

	Language A	Language B
Legal	{soʃ, sosoʃ, . . . }	{ʃoʃoʃ, ʃos, . . . }
Illegal	{*sʃo, *sosʃ, . . . }	{*soʃ, *sosoʃ, . . . }
Grammar	*sʃ: no sʃ sequence	*s. . . ʃ: no s followed by ʃ

Table 1: Sample languages A and B

A **grammar** of a language is defined as a set of constraints that predicts the wellformedness of a string—what's legal or illegal—in that language. For example, "no adjacent sʃ sequence" is the grammar for language A, and "no s can be followed by an S at any distance" the grammar for language B. The grammar for language A can be called **local** because its constraints refer only to segments that are immediately adjacent to each other. The grammar for langauge B, in contrast, is **nonlocal**, in the sense that the constraint that defines it refers to two segments at any distance.

We refer to a set of languages as a **class**. The languages sharing the local property of grammar A form a Strictly Local (SL) class, and those sharing the nonlocal property of grammar B are in the Strictly Piecewise (SP) class (Heinz & Rogers, 2010; Heinz, 2018).

* Thanks to Adam Jardine, Adam McCollum, Jeff Heinz, three anonymous reviewers, and the audience at Rutgers PhonX group, and the Language Processing Group at UC Irvine for helpful comments. Special thanks to Maria Gouskova, Gillian Gallagher, Seoyoung Kim for their help on the dataset.

Proceedings of *AMP 2021*

2.2 *Formal grammars as automata* A grammar can be formally encoded as an automaton that generates or accepts the strings in the corresponding language. In the case of phonotactics and phonology, it turns out that grammars can be modeled by **Finite-state Automata** (FSAs) (Karttunen, 1991; Kaplan & Kay, 1994). For example, here is an FSA for Grammar *sʃ:

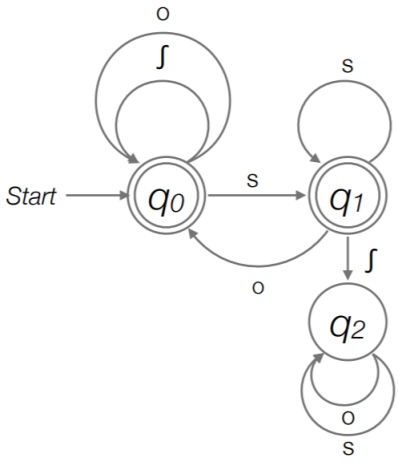

Figure 1: Sample FSA for Grammar *sʃ

A FSA consists of a set of **states** (in Figure 1, we have states q_0, q_1, and q_2) which are conventionally drawn as circles, and a set of **symbols**, which in phonotactics could be segments, features, or natural classes. In the example in Figure 1, the symbols are segments. An FSA furthermore includes a set of transitions from state to state given each symbol, which are drawn as arrows. Whether the final state is an accepting state indicates the legality of a string. If the machine enters an **accepting state** indicated by a double circle, then it is said to accept the string of symbols that it has read up to that point. If it is not in an accepting state after reading the final symbol of the string, then that string is not accepted.

In a finite-state automaton such as in Figure 1, the set membership of a string such as soʃ is determined by the following procedure. Starting in the initial state (conventionally the one labeled q_0), the automaton reads the first symbol s, and then moves to state q_1; from q_1, the automaton reads the second symbol o, enters state q_0; similarly, from state q_0, the automaton reads the third symbol ʃ, and moves to q_0. In the example, the machine ends in q_0, which is an accepting state, so soʃ is a legal string. In comparison, if the processed string is illegal, such as *sʃo, after reading the first symbol s and entering state q_1, the automaton reads an ʃ, enters and eventually stays at non-accepting state q_2.

2.3 *Subregular Hierarchy* In the study of Formal Language Theory, the Chomsky-Schützenberger Hierarchy (Chomsky & Schützenberger, 1963) demarcates the expressivity of formal grammars. Previous studies have provided evidence that most attested phonological patterns belong to a very restrictive region at the bottom of Chomsky-Schützenberger Hierarchy, known as *Subregular Hierarchy* (Heinz, 2018). The Subregular Hierarchy can be defined in terms of constraints on the structure of FSAs.

For example, in a Strictly k-Local (k-SL) language, each symbol is only subject to constraints involving itself and the immediately preceding $k-1$ symbol(s). The grammar *sʃ defines a 2-SL language. In a Strictly k-Piecewise (k-SP) language, each symbol depends on the presence of any preceding $k-1$ symbols at arbitrary distance; the grammar *s…ʃ is an example that defines a 2-SP language. SP can capture the nonlocal dependencies missed in a SL grammar. For a word [sipoʃ], in a 2-SL lnguage, [ʃ] is **only** conditional on its immediately preceding one symbol [o]; in a 2-SP language, however, [ʃ] is conditional on any preceding 1 symbol, including [s].

3 Probabilistic approach

Taking a probabilistic approach, we conceive of phonotactic learning as the process of inducing from a set of attested words a probability distribution $p(x)$, for all words $x \in \Sigma^*$. In this view, phonotactic learning is

successful when words that are considered phonotactically legal (whether attested or not in a particular dataset) receive high probability and illegal words receive low probability.[1]

In order to incorporate the formal insights described above, we propose to parameterize the phonotactic distribution $p(x)$ as a **Probabilistic Finite-state Automaton** (PFA). As illustrated in Figure 2, a PFA augments a finite-state automaton by associating each transition from state to state with a probability. Like a Finite-state Automaton, a PFA has a finite set of states $Q = \{q_0, q_1, \ldots, q_n\}$ and an inventory of symbols Σ. In addition, it has an **emission distribution** which gives the probability of generating a symbol $x \in \Sigma$ given state $q \in Q$, and a **transition distribution** which gives the probability of transitioning from state q into new state q' after the emission of symbol x.

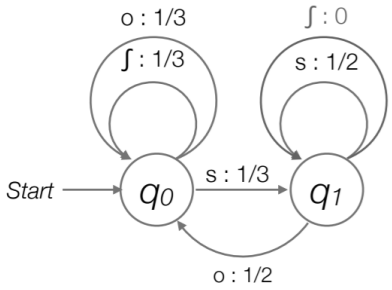

Figure 2: Sample PFA for Grammar *sʃ

The generation of a particular string $\mathbf{x} = x_1 x_2 \ldots x_n$ works by starting in an initial state q_0, generating the symbol x_1, transitioning into the next state q', and so on recursively until reaching an accepting state, where there is some nonzero probability to halt the generation process. The likelihood $p(x)$ assigned to form x is simply the probability that the PFA generates x. For example, in the PFA in Figure 2, the likelihood of an illegal word sosʃ is $\frac{1}{3} * \frac{1}{2} * \frac{1}{3} * 0 = 0$, and the likelihood of a legal word sosoʃ is $\frac{1}{3} * \frac{1}{2} * \frac{1}{3} * \frac{1}{2} * \frac{1}{3} = \frac{1}{108}$, which is higher than the likelihood of the illegal word.

A PFA can be parameterized in terms of linear algebra using a family of matrices. The **emission matrix** ($\mathbf{E}$), of shape $|Q| \times |\Sigma|$, gives the probability of emitting a symbol x given a state. Each row in the matrix represents a state, and each column represents an output symbol. Each symbol x is associated a **transition matrix** $\mathbf{T}_x$, of shape $|Q| \times |Q|$, which gives the probabilities to transition into a state q' from state q after emitting symbol x.[2] Given matrices $\mathbf{E}$ and $\mathbf{T}$, the likelihood of a form (sequence of symbols) is found by marginalizing over all trajectories through states; this calculation can be performed efficiently using a dynamic programming algorithm Vidal et al. (2005b).

Probabilistic automata to generate SL and SP languages can be implemented by hard-coding transition matrices $\mathbf{T}$ with certain structural constraints. For example, an automaton for a 2-SL language has transition matrices where each state corresponds to a symbol, and the transition matrix always forces the automaton to go into the state for symbol x after generating symbol x. In this way, the state of the machine always encodes only the previous symbol, thus generating a 2-SL language. The construction for 2-SP languages is more complex and involves building up the transition matrices $\mathbf{T}$ as a product of many smaller matrices: for details see Dai (2021) and Dai & Futrell (2021).

Using our construction, it is also possible to train an automaton with the ability to condition the generation of each symbol on both SP and SL factors, by taking the product of SP and SL automata (Heinz & Rogers, 2013). We refer to the language generated by such an automaton as SP + SL.

[1] This is distinct from a model which reads a word and outputs a binary set-membership judgment, or a model which reads a word and outputs a continuous acceptability value. For formal consequences of this passage to probabilistic automata, see Icard (2020).

[2] In fact any FSA, whether probabilistic or not, can be parameterized in this way. To encode a non-probabilistic FSA such as in Figure 1, all the entries in the matrices that define the automaton would be binary-valued, indicating whether a transition is categorically allowed or not.

4 Learning

By parameterizing our phonotactic distribution $p(x)$ as a PFA, we enable the use of a particularly simple learning algorithm. Let $p_A(x)$ denote the probability assigned to form x given a particular PFA A. The key aspect of our framework that enables simple learning is that the likelihood of a form $p_A(x)$ is a *differentiable* function of the matrices $\mathbf{E}$ and $\mathbf{T}$ that define the PFA A; it is therefore straightforward to find the matrices $\mathbf{E}$ and $\mathbf{T}$ that maximize the total likelihood assigned to a training set of attested forms using gradient descent, as we describe below. Other methods are possible (Miller et al., 2021).

Given a training set of N attested forms $\{x_n\}_{n=1}^N$, our proposed learner finds a PFA A to maximize the function $J(A)$ which is the sum log likelihood assigned to the whole training set:

$$J(A) = \sum_{n=1}^{N} \log p_A(x_n). \tag{1}$$

The function $J(A)$ is called the objective function for automaton A.

An automaton to optimize this objective function can be found by gradient descent, by starting with an initial automaton A_0 with random emission and transition matrices, and then iteratively updating the parameters of the matrices according to the following rule:

$$A_t = A_{t-1} + \eta \nabla J(A_{t-1}),$$

where η is a small scalar called the 'learning rate', the notation A_t is being used here to denote the vector of parameters that define the PFA at the t'th step of training (that is, all the numbers that populate the emission and transition matrices), and $\nabla J(A_{t-1})$ is the gradient of the objective function (1), that is, the vector of derivatives with respect to each parameter of the PFA at training step $t-1$.[3]

Gradient descent and maximum-likelihood algorithms such as ours are commonplace in phonotactic learning; for example they feature prominently in the phonotactic learner of Hayes & Wilson (2008). Furthermore, modern 'deep learning' approaches to machine learning are based almost entirely on gradient descent algorithms (Goldberg, 2017). Our contribution is to demonstrate the utility of an extremely straightforward gradient descent algorithm for PFA learning.

Using our method, it is possible to induce both the emission and transition matrices of a PFA, thus learning a PFA 'from scratch' without any predefined constraints its structure except the number of states; we call this an **unrestricted PFA**. Besides implementing such learning, we also explored the application on restricted language classes, such as SL and SP. For learning within these classes, we hard-code the transition matrices $\mathbf{T}$ to reflect the appropriate formal constraints, and we induce only the emission matrix $\mathbf{E}$ from data while holding $\mathbf{T}$ fixed. The code of the model is available at `http://github.com/hutengdai/PFA-learner`.

5 Evaluation

In this section, we show two applications of our method: first we evaluate the ability of our learner to acquire appropriate PFAs from data for toy formal languages, and then we evaluate the ability of the PFA to model real linguistic data involving nonlocal phonotactic constraints.

We use two evaluation metrics—legal–illegal difference and heldout LL—which measure different aspects of the performance of our learner. The legal–illegal difference measures the ability to represent a particular constraint; the heldout LL measures general ability to predict well-formed words. Below, when we evaluate on toy languages, we focus on the legal–illegal difference; when we come to real natural language data, we also introduce the heldout LL evaluation metric.

5.1 *Toy languages* Toy languages are defined over the symbol inventory $\{a, b, c\}$ plus the boundary symbol #. As an example of 2-SL languages, we use the language characterized by the forbidden factor **ab*.

[3] An additional complication is that we need a way to put constraints on the matrices $\mathbf{E}$ and $\mathbf{T}$ so that the rows in the matrices sum to one, i.e. so that they define a well-formed probability distribution. We accomplish this by parameterizing the emission and transition matrices using underlying real-valued matrices which are transformed into probability distributions by a softmax operation. For details see Dai & Futrell (2021).

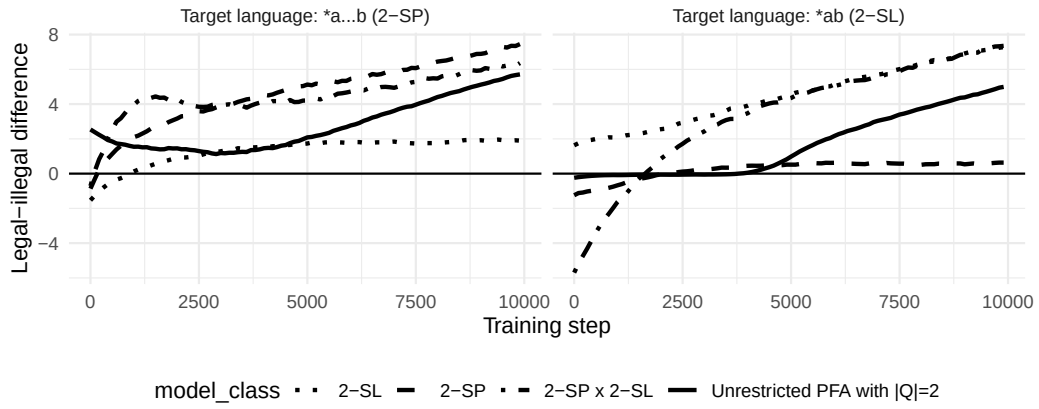

Figure 4: Difference in log probabilities for legal and illegal forms over the course of PFA induction for toy languages. A large positive value indicates that the relevant constraint has been learned.

A PFA for the language is given in Figure 3 (left). The language contains all strings that do not have an *a* followed immediately by a *b*. A PFA for a 2-SP language is shown in Figure 3 (right).

We evaluate the ability of our learner to induce these languages from data by exposing it to 10,000 sample strings generated from the reference PFAs in Figure 3, and then testing the ability of the learner to assign higher probability to a **legal test string**, which follows the restrictions of the target language, as compared with an **illegal test string**, which violates the restrictions, but is otherwise matched with the legal string in terms of factors such as length. For example, for the 2-SL toy language, our legal test string is *bacccb#* and the illegal test string is *babccc#*.

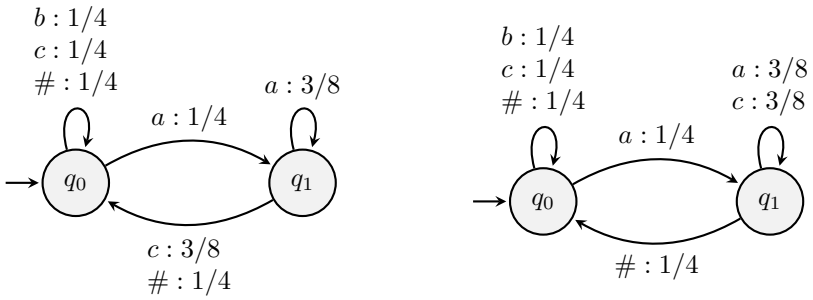

Figure 3: Reference automata for the 2-SL language characterized by the constraint **ab* (left) and the 2-SP language characterized by the constraint **a...b* (right). Arcs are annotated with symbols emitted and their corresponding emission probabilities.

For both toy languages, we test four learners: (1) an unrestricted PFA with two states, which induces its transition structure from data, (2) a 2-SL automaton, (3) a 2-SP automaton, and (4) an automaton that can represent both SL and SP restrictions (the product of a 2-SL automaton and a 2-SP automaton), which we call 2-SP + 2-SL.

Results are shown in Figure 4 in terms of the difference in log probability for the legal string minus the illegal string (the **legal–illegal difference**). A positive value indicates that the legal string is receiving higher probability. We see that, for PFAs that instantiate the appropriate language class, the legal–illegal difference increases without bound over the course of training, indicating that the relevant restrictions are being learned. For PFAs from the inappropriate language class (for example the 2-SP automaton applied to data from the 2-SL language), the legal–illegal difference levels off quickly at a small value and stops improving, indicating

failure to learn the restriction.

5.2 *Nonlocal Phonotactics in Navajo and Quechua* We now apply our proposed learner to wordlist data from real languages, Navajo and Quechua, which exhibit nonlocal phonotactic constraints (Gouskova & Gallagher, 2020). For evaluation, we measure the legal–illegal difference using artificially generated **nonce form** datasets containing (1) phonotactically legal words and (2) phonotactically illegal words which violate certain nonlocal phonotactic constraints while being otherwise well-formed and matched with the legal words in length (these datasets were produced by Gouskova & Gallagher, 2020). Given these nonce form datasets, we can compute an average legal–illegal difference.

In addition to the legal–illegal difference computed on nonce forms, we also examine the **heldout log likelihood (LL)** for forms that are attested but were not present in the training data that was given to the learner. Heldout LL is a widely-used metric in machine learning and other fields, with recent applications in linguistics (Pereira, 2000; Bishop, 2006); it is the average log likelihood of forms that are attested but not seen during learning. The heldout LL tells us how accurate the learned PFA is in general when predicting the heldout forms. A high heldout LL indicates that the PFA has learned generalizations from the training data that are useful in predicting real forms.

5.2.1 *Phenomena and data* In Navajo, the co-occurrence of alveolar and palatal strident, e.g *s…ʃ), is illegal. The learning data of Navajo includes 6279 Navajo phonological words; we divide this data into a training set of 5023 forms and a held-out set of $1,256$ forms. The testing data of Navajo consists of 5000 generated nonce words, which were labelled as illegal ($N = 3271$) and legal ($N = 1729$) based on whether the nonlocal phonotactics are satisfied.

In Quechua, any stop cannot be followed by an ejective or aspirated stop at any distance. The learning data of Quechua includes 10804 phonological words, which we separate into 8643 training forms and 2160 held-out forms. The testing data of Quechua consists of 24352 nonce forms which were manually classified as legal ($N = 18502$) and illegal ($N = 5810$, including stop-aspirate and stop-ejective pairs).

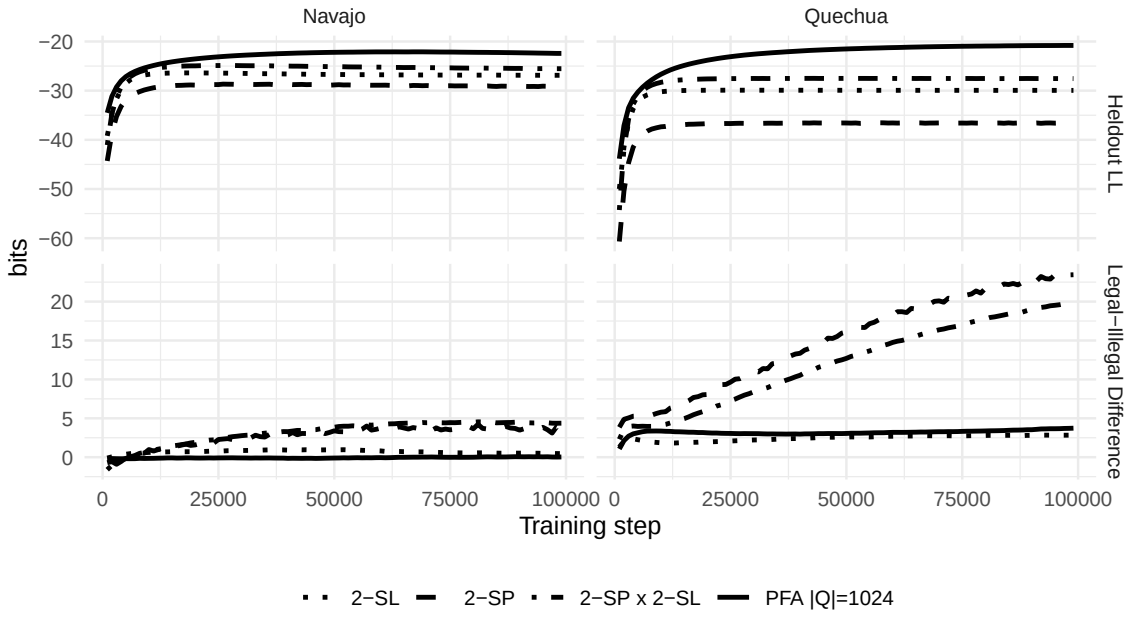

Figure 5: Performance of a 2-SP automaton, a 2-SL automaton, a 2-SP + 2-SL product automaton, and an unrestricted PFA with 1024 states

5.2.2 *Results* Figure 5 shows the performance over the course of training of a 2-SP automaton, a 2-SL automaton, a 2-SP + 2-SL product automaton, and an unrestricted PFA with 1,024 states.

Examining the Heldout LL, we find that the unrestricted PFA learner achieves the highest value, with the SP+SL learner outperforming the SL learner, indicating that more powerful automata are able to model the general phonotactic constraints of these languages. The SP learner performs the worst by this evaluation.

Turning to the Legal-Illegal Difference evaluation, we find that the SP and SP + SL learners achieve substantial legal–illegal differences. This indicates that these learners (and only these learners) successfully detected the nonlocal phonotactic phenomena in these languages.

5.3 *Discussion* When training and testing on attested phonotactic patterns, we find that the SL and SP+SL learners are able to induce the relevant nonlocal constraints, as indicated by the evaluation on nonce forms. The SP learner and the unrestricted PFA learner do not appear to detect these constraints. On the other hand, when evaluating based on heldout LL, we find that the unrestricted PFA learner is most effective in learning to predict attested but heldout forms.

The apparent discrepancy between the two evaluation metrics reflects the different natures of the evaluation. Heldout LL evaluates the ability to predict held-out forms in general, effectively measuring the ability of the model to learn *all* the phonotactic constraints from the training data. The legal–illegal difference evaluated on nonce forms, on the other hand, reflects the ability to represent only *one particular* constraint. The two metrics can be expected to differ if the particular nonlocal cosntraint probed by the nonce evaluation is only one of many constraints that are relevant for the phonotactics of the language in question.

We conjecture that the unrestricted PFA is learning many phonotactic constraints, including many local ones such as 3-SL constraints not captured by the 2-SL learner, and that these have high utility in terms of predicting unseen forms. The unrestricted PFA learner does not pick up on the nonlocal constraints because they are less useful on average in terms of predicting forms segment-by-segment. This argument suggests that a more powerful unrestricted PFA (for example one with more states) might be able to detect the nonlocal constraints.

Within the class of restricted PFAs, we find the best overall performer is the SP+SL learner, which can represent both local and nonlocal constraints. This supports the claim in Heinz & Rogers (2013) that SP + SL is advantageous for characterizing natural language phonotactics, while also suggesting that there exist other constraints beyond this class.

6 Conclusion

We implemented a differentiable framework for inducing probabilistic finite-state automata for phonotactics from data, and compared the learning of (sub)regular languages from corpus data. Our learner successfully learns nonlocal constraints when restricted to the appropriate formal language class, and achieves good performance in predicting attested forms in general. Our framework shows the utility of a probabilistic approach to phonotactics. It enables the implementation of simple learning algorithms and their evaluation on both natural and artificial data. We showed that it's possible to compare the induction of various (sub)regular languages in a unified framework. Inducing unrestricted Probabilistic Finite-state Automata (PFAs) produces the best overall fit to naturalistic held-out forms; however, a restricted subregular model (Strictly Piecewise) is superior in capturing specific nonlocal constraints as evidenced in nonce data. Models in the SP + SL language class show strength both in learning nonlocal constraints and in predicting naturalistic held-out forms.

References

Bishop, Christopher (2006). *Pattern recognition and machine learning*. Springer.

Chomsky, Noam & Marcel P. Schützenberger (1963). The algebraic theory of context free languages. Braffot, P. & D. Hirschberg (eds.), *Computer Programming and Formal Languages*, North Holland, Amsterdam, 118–161.

Dai, Huteng (2021). Learning nonlocal phonotactics in Strictly Piecewise phonotactic model. Bennett, Ryan, Richard Bibbs, Mykel Loren Brinkerhoff, Max J. Kaplan, Stephanie Rich, Amanda Rysling, Nicholas Van Handel & Maya Wax Cavallaro (eds.), *Proceedings of the 2020 Annual Meeting on Phonology*.

Dai, Huteng & Richard Futrell (2021). Simple induction of (deterministic) probabilistic finite-state automata for phonotactics by stochastic gradient descent. *Proceedings of the 18th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, Association for Computational Linguistics, 167–176.

Goldberg, Yoav (2017). *Neural network methods for natural language processing*, vol. 37 of *Synthesis Lectures on Human Language Technologies*. Morgan & Claypool, San Rafael, CA.

Gouskova, Maria & Gillian Gallagher (2020). Inducing nonlocal constraints from baseline phonotactics. *Natural Language & Linguistic Theory* 38:1, 77–116.

Hayes, Bruce & Colin Wilson (2008). A maximum entropy model of phonotactics and phonotactic learning. *Linguistic Inquiry* 39:3, 379–440.

Heinz, Jeffrey (2018). The computational nature of phonological generalizations. Hyman, Larry & Frans Plank (eds.), *Phonological Typology*, Springer, 126–195.

Heinz, Jeffrey & James Rogers (2010). Estimating Strictly Piecewise distributions. *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Uppsala, Sweden, 886–896.

Heinz, Jeffrey & James Rogers (2013). Learning subregular classes of languages with factored deterministic automata. *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, Association for Computational Linguistics, Sofia, Bulgaria, 64–71.

Icard, Thomas (2020). Calibrating generative models: The probabilistic Chomsky–Schützenberger hierarchy. *Journal of Mathematical Psychology* 95, p. 102308.

Kaplan, Ronald & Martin Kay (1994). Regular models of phonological rule systems. *Computational Linguistics* 20:3, 331–378.

Karttunen, Lauri (1991). Finite-state constraints. *Proceedings of the International Conference on Current Issues in Computational Linguistics*, Penang, Malaysia.

Miller, Jacob, Guillaume Rabusseau & John Terilla (2021). Tensor networks for probabilistic sequence modeling. *International Conference on Artificial Intelligence and Statistics*, PMLR, 3079–3087.

Pereira, Fernando (2000). Formal grammar and information theory: Together again? *Philosophical Transactions of the Royal Society* 358, 1239–1253.

Rogers, James & Geoffrey K. Pullum (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20:3, 329–342.

Shibata, Chihiro & Jeffrey Heinz (2019). Maximum likelihood estimation of factored regular deterministic stochastic languages. *Proceedings of the 16th Meeting on the Mathematics of Language*, Association for Computational Linguistics, Toronto, Canada, 102–113.

Vidal, Enrique, Franck Thollard, Colin de la Higuera, Francisco Casacuberta & Rafael C. Carrasco (2005a). Probabilistic finite-state machines – Part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27:7, 1013–1025.

Vidal, Enrique, Franck Thollard, Colin de la Higuera, Francisco Casacuberta & Rafael C. Carrasco (2005b). Probabilistic finite-state machines – Part II. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27:7, 1026–1039.