Generalizing French Schwa Deletion: the Role of Indexed Constraints

Aleksei Nazarov[1], Brian Smith
Utrecht University[1]

1 Introduction

Grammars and statistical models may suffer from two opposite problems: not capturing the training data accurately (underfitting), and fitting the training data too closely, resulting in a failure to generalize to new unseen data (overfitting). The challenge when modelling a phonological pattern is to capture the training data accurately while maintaining the ability to generalize. Lexically-specific mechanisms like indexed constraints (Pater 2000) and cophonologies (Inkelas and Zoll 2007) are designed to allow the grammar to better capture the intricacies of a dataset that involves phonologically unexplainable differences between morphemes. The question addressed here is: will giving the grammar the power to encode phonological patterns on a morpheme-by-morpheme basis hurt its ability to generalize? We will show that, at least for the case study addressed here, it does not.

We focus on the case of French schwa deletion (§2), an optional process whose rate of application is modulated by both phonological and lexical factors. Each word has its own rate of deletion, which means that an indexed constraint grammar could heavily rely on indexed constraints over general phonological constraints, hampering its ability to generalize to novel words. In fact, constraint indexation theoretically allows patterns to be stored for every individual lexical item. It might turn out that there is little work left to do for the non-indexed constraints, which will cause generalization to suffer. Thus, adding indexed constraints to the grammar could lead to significant overfitting, but not adding indexed constraints would lead to underfitting, as the lexical factors are not taken into account. Is a balance between lexical factors and generalizability possible?

We tested the trade-off between overfitting and underfitting by comparing various Maximum Entropy (MaxEnt; Goldwater & Johnson 2003) grammar learners: one that only assigns weights to non-indexed, lexicon-wide constraints, and three novel learners that induce lexically indexed constraints and weight them alongside their non-indexed counterparts. These four learners were trained on probabilities of French schwa deletion within words (taken from the experimental results in Racine 2008), and then tested on a completely different dataset: probabilities of French schwa at word boundaries (taken from the experimental results of Smith & Pater 2020). This allows us to test both underfitting (how well does the grammar capture the data in the corpus?) and overfitting (how well does the grammar generalize to the new dataset?). We find that adding indexed constraints indeed does decrease underfitting, while most indexed constraint learners do well on generalizing to the new dataset, and the best indexed constraint learners do not increase overfitting.

The rest of this paper is organized as follows: §2 will present French schwa deletion, including both datasets (training and testing); §3 will present the indexed constraint learners that we propose in this paper; §4 will describe how the learners were applied to the training data and the results of this, while §5 will describe the generalization to the test data. Finally, §6 will offer some discussion and our concluding remarks.

2 French schwa deletion

We apply our indexed constraint induction learners, which will be described in more detail in §3, to the

* Many thanks to Isabelle Racine for generously allowing us to use her data, and to Outi Bat-El, Noam Faust, Gaja Jarosz, Brandon Prickett, and audiences at the 29th Manchester Phonology Meeting and at IATL 37 at the Hebrew University of Jerusalem for their useful feedback and comments. All errors are our own.

well-known case of deletion of schwa in French (see, e.g., Dell 1985), a phonological process whose rate of application is conditioned by phonological context (see, e.g., Racine 2008, Smith & Pater 2020 for a review). Schwa deletion is a good test case because it is both optional and lexically-conditioned. The examples in (1a) demonstrate optionality: probabilistic process application, where rate of application depends on context, while the examples in (1b) demonstrate lexical-conditioning: application rates differ between words, even when controlling for phonological conditioning.

(1) a. *deletion is optional*
/la səmɛn/ → [lasmɛn, lasəmɛn] 'the week' (Dell 1985)
b. *probability of deletion depends on specific word*
/səme/ → [səme], *[sme] 'to sow' /səmaj/ → [səmaj, smaj] 'sowing' (Dell 1985)
/səmestʁ/ → [smestʁ] 'semester' *less likely than* /səmɛn/ → [smɛn] 'week' (Racine's 2008 data)

These properties make the French case ideal for testing a probabilistic learner that builds a grammar with indexed constraints: the learner should be able to capture the data it is trained on, including lexical conditioning, while also being able to generalize the patterns therein to novel words. To achieve this, we train the learner described in §3 on a dataset taken from Racine (2008), and then test the resulting grammars both on this dataset and on experimental results taken from Smith & Pater (2020). We will describe both datasets, including their embedding in the grammar, in §2.1 and §2.2, respectively.

Before discussing the dataset, we will go through the phonological conditions known to govern French schwa deletion, and we model the dataset and the constraints that enforce them. We will focus on morpheme-internal schwas, since schwas at morpheme boundaries are often analyzed as epenthetic (e.g., Dell 1985, Côté 2000; see Smith & Pater 2020 for an overview).

The surrounding segmental context is known to have an effect on the probability of schwa deletion (Dell 1985, Côté 2000, Racine 2008). Firstly, if schwa deletion leads to a three-consonant cluster, this is less likely than if it leads to a two-consonant cluster, as in (2a). Furthermore, among examples where three-consonant clusters as a result of schwa deletion are acceptable, the likelihood of deletion is modulated by the sonority of the consonants. For instance, as pointed out by Côté (2000) and Kaplan (2011), clusters whose medial member is most sonorous are avoided, as in (2b), and clusters that end in a stop followed by a non-approximant (= a nasal or an obstruent) are also avoided to an extent, as in (2c).

Finally, Dell (1985) reports that among cases where deletion yields two-consonant clusters, the schwa deletion rate is lower if this is a word-initial cluster as opposed to a word-medial cluster, as in (2d), where the schwa in /avəniʁ/ may not be deleted. In fact, Dell's analysis posits that schwa deletion is obligatory in within-word VC_ environments and optional in V#C_ environments.

Schwa deletion is also conditioned by prosodic factors (Dell 1985): there is more deletion when schwa is followed by multiple syllables in the phrase compared to when it is followed by just one syllable; see (2e).

(2) a. *less deletion when it leads to CCC (examples from Dell 1985)*
/mɑ̃ʒ lə gato/ → [mɑ̃ʒləgato], *[mɑ̃ʒlgato] 'eat (sg.) the cake'
/mɑ̃ʒe lə gato/ → [mɑ̃ʒeləgato, mɑ̃ʒelgato] 'eat (pl.) the cake'
b. *less deletion when resulting CCC cluster has highest sonority in the middle (ex.: Côté 2000)*
/ɑ̃vi də tə lə dəmɑ̃de/ → *[ɑ̃vidətldəmɑ̃de] 'feel like asking you'
/ɑ̃vi də tə lə dəmɑ̃de/ → [ɑ̃vidtlədəmɑ̃de] 'feel like asking you'
c. *less deletion when resulting CCC cluster ends in a stop then a non-approximant (Côté 2000)*
/ɑ̃vi də tə lə dəmɑ̃de/ → ??[ɑ̃vidətəldmɑ̃de] 'feel like asking you'
/ɑ̃vi də tə lə dəmɑ̃de/ → [ɑ̃vidtlədəmɑ̃de] 'feel like asking you'
d. *less deletion when resulting cluster is word-initial (examples from Dell 1985)*
/la səmɛn/ → [la#səmɛn, la#smɛn] 'the week'
/avəniʁ/ → *[avəniʁ], [avniʁ] 'future'
e. *less deletion when schwa is in penult position in the phrase (examples from Dell 1985)*
/la tɛʁ sə vɑ̃/ → [latɛʁsəvɑ̃], *[latɛʁsvɑ̃] 'the land is selling'
/la tɛʁ sə vɑ̃ bjɛ̃/ → [latɛʁsəvɑ̃bjɛ̃, latɛʁsvɑ̃bjɛ̃] 'the land is selling well'

To capture these generalizations and model deletion we use the constraint set in (3). The constraints *ə and

MAX are used to drive and prevent deletion, respectively (as in Smith & Pater 2020). Differences between their relative weights can capture different baseline deletion rates between words. *CCC captures the fact that deletion is less likely when it results in a three-consonant cluster (2a). The constraints *CNC and *CTN (from Kaplan 2011, based on Côté 2000) are motivated by the observation that these types of three-consonant clusters are particularly avoided as the outcome of schwa deletion, as in (2b,c). *#CC captures the relative dispreference for schwa deletion that leads to a word-initial cluster, as in (2d). Finally, the [ə]→penult constraint, which penalizes schwa outside of the penultimate syllable position, is motivated by the observation that schwa is deleted less often (and thus, tolerated more often) in penultimate syllables, (2e).

(3) *Constraint set*
*ə: One violation mark for every instance of [ə].
MAX: One violation mark for every deleted segment.
*CCC: One violation for every sequence of three consonants (Cs).
*CNC: One violation for every sequence of three Cs, in which the middle one is the most sonorant.
*CTN: One violation for every sequence of a C, then a plosive (=T), then a nasal or obstruent (=N).
*#CC: One violation for every word-initial consonant cluster.
[ə]→penult (=ə→PU): One violation mark for every instance of [ə] in a non-penultimate syllable.

2.1 *Training data* Our training data come from Racine's (2008) rating experiments, in which participants rated schwa-ful and schwa-less pronunciations for 2112 French words (all nouns). The schwa-ful and schwa-less versions of these words were presented orthographically, and participants estimated the frequency of pronunciation on a 1-7 scale. Since MaxEnt grammars need counts or probabilities as inputs, we used the averaged ratings to estimate probability of schwa pronunciation for each word. Racine's study included both Swiss French and France French speakers, and we only consider data from the France French speakers here. We transformed ratings into probabilities by subtracting 1 from each averaged rating (so the lowest score is 0), and then dividing each adjusted rating by the sum of adjusted ratings for both variants of the word, as in (2). For instance, for the word /səmɛstʁ/ 'semester', the schwaless variant [smɛstʁ] has an averaged rating of 1.92, while the faithful variant [səmɛstʁ] has a rating of 6.50. The adjusted ratings are 0.92 and 5.50, respectively, and the resulting probability of [smɛstʁ] is 0.92/(0.92+5.50) = 0.14 = 14%. If both variants of a word have equal ratings, they will each have a probability of 50%, while if one variant has a rating of 7 and the other has one of 5, or if one variant is rated 4 and the other, 3, this would yield a 60% probability for the higher-rated word: 6/(6+4) = 3/(3+2) = 0.6 = 60%.

(4) *Transformation from ratings to estimated probabilities.*
$$P(candidate) = \frac{Rating(candidate)-1}{Rating(candidate)-1+Rating(other\ candidate)-1}$$

Of the words in the experiment, we excluded words with potential word-internal morpheme boundaries (to exclude any influence of morphological structure; for instance, we cannot assume that schwa at a boundary is underlying, see above) and words that were not transcribed as having variable schwa but were tested because the orthographic symbol for schwa, <e> in an open syllable, is present in their spelling (e.g., <casserole> /kasʁɔl/ 'pot'). In terms of morpheme boundaries, we removed words containing a dash (which indicates compound words) and those ending in <-té>, <-rie>, <-ment>, or <-mentation>. This procedure yielded 456 of the original 2112 words. We made these into OT-style tableaux with each word as an input and two candidates per input: a faithful candidate and a candidate with the schwa removed (e.g., /səmɛstʁ/ → <[səmɛstʁ], [smɛstʁ]>). Each tableau contains the seven non-indexed constraints shown in (3). These data will be used as training data in the simulations described in §4.

2.2 *Generalization data* Our trained grammars were further tested using data from Smith & Pater (2020), who studied phonological conditions on the presence or absence of schwa for both underlying and epenthetic schwas. We use the schwa deletion contexts from Smith & Pater, which consist of a C- or V-final word followed by the clitic /tə/ 'you', followed by a CVC or a CVCV word, as in (5).

(5) V-final + CVC: [eva t(ə) ˈʃɔk] ‘Eva shocks you’
V-final + CVCV: [eva t(ə) ʃɔˈkɛ] ‘Eva shocked you’
C-final + CVC: [moris t(ə) ˈsit] ‘Maurice cites you’
C-final + CVCV: [moris t(ə) siˈtɛ] ‘Maurice cited you’

In Smith & Pater’s experiment, French native speakers from France were asked to choose between two versions of the same phrase, with and without schwa. The stimuli were presented orthographically, with unpronounced schwa represented with an apostrophe, e.g., *Eva t’choque* (an orthographic representation of the first example in (5)). Statistical analysis found that, as expected, schwa is more likely to be pronounced when its absence would create a CCC cluster (C-final condition) and when the schwa is in a syllable that precedes a final stressed syllable (CVC condition). These trends are also visible in the proportions of no deletion choices within the schwa deletion condition results, illustrated in Table 1. When we test our learner’s capacity to find generalizable grammars with indexed constraints, we aim to match these probabilities with the grammars - see §5 of this paper.

Table 1. *Proportions of schwa retention choices among all schwa deletion trials.*

Context	V-final + CVC	V-final + CVCV	C-final + CVC	C-final + CVCV
Schwa retention	65%	56%	94%	91%

What makes these data so ideal for testing generalization of indexed constraint grammars is that, despite the preceding and following context consisting of various actual words, these words contain no schwa themselves, while the word that contains schwa is always the clitic /tə/, so that, whichever lexical factors may influence the realization of schwa in this word, these are constant between the conditions in Table 1, and the difference between these conditions is made by contextual factors.

3 Learner

Our learning proposal builds on existing learners that expand OT grammars with indexed (lexically-specific; see Kraska-Szlenk 1995, Pater 2000) constraints (Pater 2007, 2010, Becker 2009, Round 2017, Nazarov 2021). These existing learners start with a set of pre-specified unranked OT constraints and a dataset including attested and unattested candidates. While the constraints are being ranked, the learner creates new indexed constraints based on the pre-specified constraints as needed and includes these in the ranking. These learners also find which lexical items (Becker 2009) or underlying segments (Round 2017, Nazarov 2021) should be indexed to each indexed constraint. However, these existing proposals are intrinsically connected to Constraint Demotion (Tesar 1995) approaches to learning, which require non-probabilistic OT and cannot deal with variation. This is a problem for cases like French schwa deletion (§2), in which lexical differences lie in rates of variation between retention and deletion of underlying schwa.

In this paper, we propose several extensions of these approaches in the well-known Maximum Entropy framework (MaxEnt; e.g., Goldwater & Johnson 2003), which can easily deal with variation. Moreover, MaxEnt is compatible with various types of general-purpose optimization, since it is a type of regression model (Manning & Schütze 1999). We will first review the mechanics of the previous approaches in §3.1, and then explore the details of our proposal for indexation in MaxEnt in §3.2. Finally, two ways in which indexed constraints can be generalized to novel inputs are described in §3.3.

3.1 *Contrast as a trigger* The existing approaches (Becker 2009 and others) are based on the idea that indexed constraints are needed whenever the pre-specified constraints are insufficient to account for the contrasts in the data. In Constraint Demotion approaches, this situation is signaled by so-called inconsistency detection. Inconsistency occurs when different data points require mutually exclusive rankings, for instance, if one data point (winner-loser pair), /səmɛn/ → <[smɛn], *[səmɛn]>, requires *ə >> MAX while another, /səmɛstʁ/ → <[səmɛstʁ], * [smɛstʁ]>, requires MAX >> *ə. This can be seen as a case of irreducible contrast: there is nothing in the phonological grammar that can make the difference between the two data points (schwa deletion versus schwa retention), so there must be contrast between two inputs as to their schwa deletion status. Whenever this occurs, the existing algorithms add an indexed version of one of the constraints

involved in the inconsistency, and ensure that only the data points that require a high ranking of this constraint are indexed to it. For instance, in the MAX versus *ə case, /səmɛstʁ/ may be indexed to a newly formed indexed constraint MAX$_i$, so it will not undergo schwa deletion, and the ranking MAX$_i$ >> *ə >> MAX may be established, which leads to schwa retention in the word indexed *i*, but schwa deletion otherwise.

3.2 *Extension to MaxEnt* The categorical approach in Becker (2009) and others does not work well with variation. In fact, if a single input has two winning outputs, this also leads to inconsistency detection: /səmɛn/ → <[smɛn], *[səmɛn]> requires *ə >> MAX while /səmɛn/ → <[səmɛn], *[smɛn]> requires MAX >> *ə. Furthermore, inconsistency detection is a part of a custom-built learner for categorical OT grammars and cannot easily be extended to more general-purpose learners. To tackle both issues, we propose here an extension of the existing approaches to MaxEnt.

Instead of using inconsistency detection in ranked constraint grammars, we propose that contrast, and therefore indexed constraints, can be found by looking at the weights of a MaxEnt grammar and their gradients (the degree to by which increasing or decreasing the weight would improve the fit to the dataset or a given data point; informally, the degree to which the weight "wants" to be raised or lowered). If a constraint has a positive gradient value (↑) for some inputs and a negative gradient value (↓) for some other inputs, we can see this as a form of contrast, since there are inputs that want a higher weighting for a constraint (e.g. /səmɛn/ for MAX), and other inputs that want a lower weighting for the same constraint (e.g. /səmɛstʁ/ for MAX). Of course, this does not give us a firm conclusion on whether this leads to mutually exclusive rankings/weightings (like *ə >> MAX versus MAX >> *ə), cf. Nazarov (2018) on "soft inconsistency" in probabilistic ranking, but various approximations can be used to create MaxEnt grammars with indexed constraints. These approximations are of increasing sophistication (§3.3.1-3), and our learning simulations (§4,5) will show how these compare to one another in terms of performance.

3.2.1 *Pre-training indexation* Possibly the simplest way to approach gradient-based indexation is the following: it is assumed that each pre-existing constraint has exactly one indexed version, and before the weights of the MaxEnt model are trained on data, it is determined which inputs are connected to each indexed constraint. For each indexed constraint, its associated inputs are exactly those inputs that have a positive gradient on its non-indexed counterpart. Gradients are computer based the initial weighting of the constraints, in our case, 0 weight for all constraints. For instance, for the indexed version of MAX assumed by this algorithm, all inputs in which schwa retention is preferred (↑ MAX) will be indexed to it. For our data, there were two candidates for every input: one with and one without a schwa in a designated place, e.g., /səmɛn/ → <[smɛn], [səmɛn]>. Whenever the candidate preferred by a constraint for input X has more than 50% probability, this constraint will have a positive gradient for input X given 0 weights. For instance, if /səmɛstʁ/ has a >50% probability of being realized as [səmɛstʁ], then MAX will have a positive gradient for /səmɛstʁ/, and this input will be in the set of inputs indexed to MAX$_i$.

In practice, this approach did not work very well, since many inputs had a distribution of schwa retention/deletion close to 50-50, and inputs that happened to have the slightest preference for retention or deletion were also assigned to the indexed versions of constraints, which led to a great amount of noise in the system. To give this simple pre-training indexation learner a better chance of success, we added a threshold requirement: only inputs with at least 60% probability for one of its candidates can be associated with and indexed constraint. For instance, for MAX$_i$, /səmɛstʁ/ with only 14% schwa deletion is indexed to it, but not /səmɛn/ with 50% schwa deletion. The effectiveness of this approach will be considered in §4-5.

The advantage of this approach is its simplicity: it simply adds one indexed constraint for every pre-existing non-indexed constraint, and it requires only one round of training the grammar weights. However, one major drawback is that interactions between constraints are not taken into account. For instance, in a word like /bʁəvɛ/ 'patent, certificate', schwa deletion practically never occurs (the estimated deletion rate is 3% in the training data). This means that it will have a positive gradient for MAX and will be indexed to MAX$_i$ (which is indeed the case in our results for this learner), but this does not mean that the lower deletion rate has no phonological explanation: it can (at least partially) be explained by a high weighting of *CCC and *CNC (since deletion yields the cluster [bʁv], which contains three consonants, of which the middle one is the most sonorous). In other words, it is predicted that this approach will have a high false alarm rate for indexation, especially in context-insensitive constraints like MAX: there are independent phonological factors that could explain why /bʁəvɛ/ has a very low deletion rate, but these factors have no impact on whether this

input will be included in an exceptional high-weighted variant of the general anti-deletion constraint MAX.

To better deal with this aspect of indexation and only index inputs to indexed constraints if there is no purely phonological explanation for differences in indexation, we propose the post-training indexation learner, which waits until after the weights of the non-indexed constraints have been optimized to start inducing indexed constraints.

3.2.2 *Post-training indexation* In this "upgrade" of the pre-training learner, everything remains the same, except that gradients are computed after the weights of the pre-specified constraints have been optimized by training the grammar on the dataset: hence the name "post-training indexation". For instance, the inputs indexed to MAX$_i$ are determined after the optimal weights of MAX and all other pre-specified constraints are found. As indicated at the end of the previous section, the goal of this is to more efficiently determine which inputs are indexed to which constraint. This is done by factoring out phonological influences: giving the pre-specified constraints their optimal weights allows for phonological effects to shape output candidate probabilities. For instance, by acknowledging that *CCC and *CNC have a relatively high weight, the very low schwa deletion rate in /bʁəvɛ/ can be accounted for without recourse to indexing it to higher-weighted MAX$_i$.

In this version of the algorithm, no thresholding function is used for determining which input goes with which indexed constraint: any input that has a positive gradient for a constraint (e.g., MAX) is connected to its indexed version (e.g., MAX$_i$).

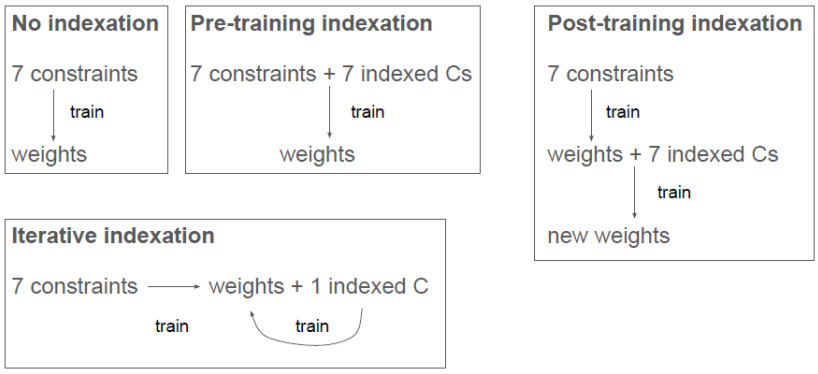
After the weights of the pre-specified constraints are optimized and the violations of their indexed versions are determined (by finding which inputs are associated with each of them, as indicated above), the pre-specified and indexed constraints are added to the same grammar and their weights are optimized again on the same data. The starting weights for this second training step are the weights found by the previous training step; indexed constraints start out as the same weight as their non-indexed counterparts. For instance, if MAX receives the weight of 1.14 after the first training step, then both MAX and MAX$_i$ start out at 1.14 for the second training step; after the second training step, the weight of MAX lowers to 1.11, while the weight of MAX$_i$ raises to 1.20.

While this learner can account for interactions between phonology and indexation, it does not allow access to interactions among indexed constraints. For instance, if an input is indexed to *CCC$_j$, it might not also need to be indexed to MAX$_i$ to account for its lower schwa deletion rate, and *vice versa*, since both constraints can lower the same word's deletion rate. Furthermore, this learner cannot choose which indexed constraints to include in the first place, and does not allow for multiple indexed versions of the same constraint. All these issues are addressed in the third learner we describe here: the iterative indexation learner, which adds indexed constraints one by one.

3.2.3 *Iterative indexation* The final "upgrade" to the learner we suggest is that, instead of adding all indexed constraints at once, only one indexed constraint is chosen, and then this modified form of the post-training indexation learner is iterated until the grammar stops improving in terms of its fit to the training data. This learner aims to induce only those indexed constraints that are strictly necessary for the training data.

The learner starts out with non-indexed constraints only, like the post-training indexation learner does. After this, a criterion is used to select one indexed constraint to add to the grammar. Just like in the other two learners, the inputs associated with this indexed constraint are those for which the non-indexed counterpart's weight's gradient is positive. Just like in the post-training indexation learner, the grammar's weights are trained on the same dataset again once the new indexed constraint has been added (and the new indexed constraint's weight is initialized at the same value as the constraint it is derived from). If the new and the old grammar differ in KL-divergence (Kullback & Leibler 1951) between predicted probabilities and the attested probabilities in the training data by more than a threshold value (we used 1), another new indexed constraint is added, and the grammar's weights are trained once again. When the new grammar no longer yields a sufficient improvement in KL-divergence (<1 in our case), the cycle stops. Figure 1 visualizes all learners together to show how iterative learner builds on the pre- and post-training learners.

Figure 1. *A visualization of the three indexation learners and the control (no indexation).*



The criterion by which each new indexed constraint is chosen is the mean absolute error (MAE). For every constraint in the grammar, the gradient values given each individual input are computed, as well as the mean of all these values. Within each constraint, the absolute differences between each value and the corresponding mean are computed. These differences are summed and divided by the total number of inputs:

(6) *Mean absolute error calculation*

$$\frac{\sum|current\ input\ gradient - mean\ gradient|}{number\ of\ inputs}$$

At any point in time, for a constraint to have an indexed version to be added to the grammar, the constraint must not already have an indexed version in the grammar and must have both positive and negative gradient values among the inputs. From the set of constraints that meet these criteria, the one with the highest MAE is chosen (i.e., the one for which the various inputs diverge the most in terms of whether they want the constraint's weight to go up of down). If there is a tie between multiple constraints that all have the same MAE, one of these is chosen at random.

This learner takes interactions between indexed constraints into account, as each indexed constraint has the chance to receive its optimized weight before the next indexed constraint is induced. For instance, all inputs indexed to *CCC$_j$ in the results of the post-training indexation learner are also indexed to MAX$_i$; the iterative indexation learner induces MAX$_i$ first, and since this sufficiently decreases the relevant inputs' probability of schwa deletion, *CCC$_j$ is never induced.

In addition, it allows for inducing multiple indexed constraints for the same non-indexed counterpart. At any iteration in the iterative indexation learner, any constraint is eligible for being indexed as long as it does not already have an indexed version in the grammar and as long as it has inputs with positive gradients and inputs with negative gradients. This means that existing indexed constraints can be re-indexed, as happens in our existing results: the indexed constraint *ə$_j$ receives an indexed version *ə$_{j,m}$ – which is associated with a proper subset of the inputs that belong to *ə$_j$. Thus, the non-indexed constraint *ə corresponds to two different indexed constraints, and the grammar encodes three levels of schwa-avoidance in the language: a base-level (only subject to *ə), a middle level (subject to *ə and *ə$_j$), and a high level (subject to *ə, *ə$_j$, and *ə$_{j,m}$).

3.3 *Generalization methods* Based on the outcomes of three different indexed constraint learners, we also explore two different ways of generalizing the resulting indexed constraint grammars to novel inputs. The most economical way, which we call the *zero method*, is to ignore indexed constraints entirely when applying the grammar to novel inputs (cf. Pater 2000). Essentially, this assumes that novel inputs cannot have indices and speakers do not use their implicit knowledge of the lexicon to generalize.

An alternative method, which we call the *probabilistic method*, models the speaker's lexical knowledge by assigning indices to novel words probabilistically (see, e.g., Becker 2009). In Becker's (2009) original proposal, a novel input is probabilistically assigned constraint indices according to the proportion of existing words that has each of these indices (i.e., if 60% of all words is indexed to MAX$_i$, then there is a 60% chance that a novel word will also receive index *i*). Here, we simplify this procedure by giving novel words "expected" violation amounts for every indexed constraint. This is done by multiplying the number of

violations of the non-indexed counterpart of the relevant indexed constraint by the proportion of relevant inputs (=inputs whose outputs candidates are not tied in terms of violations of this constraint) in the training data that are indexed to it. For example, if 60% of all inputs in the training data that can have a MAX violation are indexed to MAX$_i$, then for a novel input /bəla/, the candidate [bla] will have 60% × 1 = 0.6 violations, while the candidate [bəla] will have 60% × 0 = 0 violations. Thus, probabilities assigned to surface candidates derived from novel inputs reflect how Becker (2009) predicts inputs with the same violation profile to surface in aggregate.

4 Simulations and performance on training data

The three indexation learners described in §3.2 were implemented in R using Staubs' (2011) implementation of batch Maximum Entropy learning. Staubs' original learner was used for the control (no indexation). Each learner was run once, with a minimal L2 regularization term with μ=0 and σ^2=10,000,000. Weights were initialized at 0, and optimization took place through the L-BFGS-B (Byrd et al. 1995) batch optimization method. The four resulting grammars (no, pre-training, post-training, and iterative indexation) are summarized in Table 2. Non-zero weighted indexed constraints are shaded dark grey, non-zero-weighted non-indexed constraints are shaded light grey. The third column under each learner indicates the percentage of relevant inputs (i.e., inputs that have a non-zero violation profile) associated with each indexed constraint.

Table 2. Grammars learned by each of the learners.

No indexation			**Pre-training indexation**			**Post-training indexation**			**Iterative indexation**		
Constr	Weight	%	Constr	Weight	%	Constr	Weight	%	Constr	Weight	%
*CNC	1.56	100%	MAX$_i$	1.25	87%	*CNC	1.80	100%	*CNC	1.88	100%
MAX	1.14	100%	*CNC	0.79	100%	MAX$_i$	1.20	54%	MAX$_i$	1.83	54%
*CCC	0.92	100%	*CNC$_j$	0.79	100%	MAX	1.11	100%	MAX	1.22	100%
ə→PU	0.29	100%	*ə$_k$	0.65	1%	*CTN	0.78	100%	*CCC	0.89	100%
*CTN	0.26	100%	MAX	0.47	100%	*CCC	0.61	100%	*ə$_j$	0.78	46%
*ə	0.004	100%	*CCC	0.289	100%	*CCC$_j$	0.46	60%	*CTN	0.76	100%
*#CC	0.00[1]	100%	*CCC$_m$	0.289	97%	*ə$_k$	0.30	46%	*ə	0.56	100%
			ə→PU	0.287	100%	*ə	0.30	100%	*#CC$_k$	0.48	47%
			*ə	0.22	100%	ə→PU	0.23	100%	*ə$_{j,m}$	0.32	22%
			*CTN	0.15	100%	*CNC$_m$	0.06	57%	ə→PU	0.31	100%
			*CTN$_n$	0.15	100%	*#CC	0.00	100%	*#CC	0.00	100%
			*#CC	0.00	100%	*CTN$_n$	0.00	40%			
			*#CC$_p$	0.00	73%	*#CC$_p$	0.00	44%			
			ə→PU$_q$	0.00	1%	ə→PU $_q$	0.00	53%			

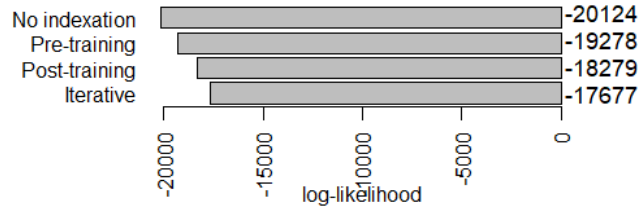
As can be seen, the iterative indexation learner indeed only induced four indexed constraints, corresponding to only three non-indexed counterparts (MAX, *ə, *#CC). In addition, it is interesting to see that the pre-training learner leads to indexed constraints that cover either a great majority of relevant inputs

[1] Dell uses pairs like /avəniʁ/ (obligatory deletion) vs. /la səmɛn/ (optional deletion) to motivate the generalization that schwa is less likely to delete when it results in a #CC sequence. In our data, there are very few words like /avəniʁ/ (just four words with schwa in a VC_CV context), so differentiating between the contexts is not necessary for the grammar. This is most likely the reason why *#CC consistently receives 0 weight.

(up to 100%) or almost none (1%), while the post-training and iterative learners yield indexed constraints with percentages closer to 50% (note that for the grammar made by the iterative indexation learner, the constraint *$ə_{j,m}$ covers roughly half of the inputs of the constraint it is based on, namely, *$ə_j$ – 22% versus 46%).

These grammars were evaluated by computing the log-likelihood of the training data given their constraints and weights. Since this involves the original training data, the generalization methods are not relevant: the resulting tableaux files specify for each input whether it is associated with a certain indexed constraint. The results of this are given in Figure 2. It can be seen that each of the indexed constraint grammars is captures the data better than the grammar without indexed constraints, and that the log-likelihood goes up further as the indexation learners become more sophisticated (pre-training < post-training < iterative). Note that sophistication does not equal model complexity in this case, since the iterative learner's grammar only has 11 constraints against 14 constraints in the pre-training and post-training learners' grammars. Furthermore, if we correct for zero-weighted constraints, the pre-training learner's grammar actually has the greatest number of constraints: 11, versus 10 for the post-training and iterative learners.

Figure 2. *Log-likelihood on training data for all four grammars (closer to 0 is better).*



5 Generalization to Smith & Pater (2020)

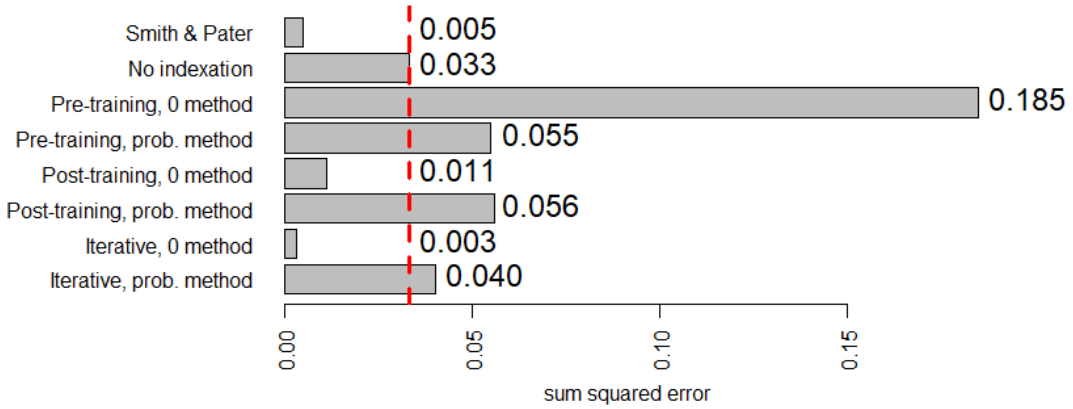
To see whether the grammars learned by each of the indexation learners generalized well to new data, we tested the grammars resulting from all four learners on a new data set. Specifically, the constraints and their weights in these grammars, as trained on the Racine (2008) corpus (§2.1, §4), were used to predict the schwa retention rates from Smith & Pater's (2020) experiment (see §2.2). We made generic inputs /VCəCV/, /VCəCVCV/, /CCəCV/, and /CCəCVCV/, corresponding to each of the conditions in the experiment, and gave them two candidates each: one with schwa, one without. Only some of the constraints and their weights in the learned grammars were used: *ə, ə→PU, Max, *CCC, *CTN, and any indexed versions of these. *CTN was violated whenever *CCC was, since the experimental items all has a plosive preceding schwa and an obstruent following it. *#CC was irrelevant due to a lack of initial clusters in these data, while *CNC was irrelevant because all CCC clusters has a stop as its middle consonant.

Both generalization methods explained in §3.3 (zero method, probabilistic method) were applied to the outcome of each of the three indexation learners (pre-training, post-training, iterative indexation). The no indexation grammar has no indexed constraints, and thus the generalization methods are irrelevant to it. This means there are seven distinct combinations: no indexation, and then each indexation model combined with either generalization method. Each grammar's weights as applied to the generic inputs yield predicted candidate probabilities; the predicted probabilities of the schwa-ful candidates were compared to the attested proportions of schwa retention in the experiment. The comparison was done through sum squared error (SSE): the differences between the predicted and attested proportions of schwa retention for each condition were squared and then summed, see (5). Lower SSE means the prediction is more accurate. Figure 3 shows the SSE values for each learning model × generalization method combination, with the SSE value of the Smith & Pater's (2020) own batch-trained MaxEnt grammar as a reference. This latter grammar had essentially the same set of constraints relevant to the deletion subset of the data, but did not have *CTN, and had *CLASH instead of ə→PU; it also had no indexed constraints (see Smith & Pater 2020 for details). The dotted line in Figure 3 indicates the SSE value of the no indexation grammar from §4.

(7) *SSE calculation.*

$$SSE = \sum (predicted - attested)^2$$

Figure 3. *SSE values for each combination. Dotted line indicates No indexation model SSE value.*



As can be seen in Figure 3, the learner × generalization method combinations differ in their error values, but there is one clear outlier: the pre-training indexation learner combined with the zero method of generalization scored exceptionally badly. Upon inspection of the constraints in the learned grammar (Table 2), it can be seen that many of the indexed constraints (including most of the highly weighted ones) are associated with a great majority of the inputs in the training data, so that indexed constraints play a crucial role in deriving the correct schwa deletion rates for most inputs. This means that applying the zero method (i.e., ignoring the indexed constraints) will yield very different rates of schwa deletion compared to the ones in the training data, meaning that schwa retention/deletion rates in novel data points will also be predicted incorrectly.

Interestingly, while the probabilistic method yields a better score for the pre-training learner's outcome, the pattern is the opposite for the post-training and iterative indexation learners: when the 0 method is applied, the outcome of these learners fits the test data much better compared to when the probabilistic method is applied. This is probably related to the fact that these learners lead to much sparser use of indexed constraints in fitting the training data (i.e., lower percentages of relevant inputs being associated with any given indexed constraint), so that leaving out indexed constraints better reflects the tendencies in the majority of the data.

The SSE values for all combinations except pre-training indexation with the 0 method of generalization are quite close to that of the no indexation model (they fall close to the dotted line). In particular, the best values (for post-training and iterative indexation with the 0 method of generalization) are on par with the no indexation model: their SSE values do not fall out higher. The iterative indexation grammar with the 0 method of generalization is even on par with Smith & Pater's own grammar, as both have similar SSE values.

The main takeaway is that adding indexation does not hurt generalization. Even though indexation could in theory create an overfitted grammar where patterns are stored for every lexical item, in practice, learning algorithms for indexation result in grammars in which non-indexed constraints can be used to generalize to previously unseen data.

6 Discussion/conclusion

Adding indexed constraints to a constraint-based grammar is expected to lead to better capturing the intricacies of the dataset, but in this paper, we ask whether this hurts the ability of these grammars to generalize to novel inputs. We used two existing French schwa deletion data sets as a case study: one for training (§2.1), one for testing (§2.2). These datasets showcase probabilistic patterns that are both phonologically and lexically-conditioned, which is a good match for our proposal of MaxEnt grammar learners that induce indexed constraints (§3). As can be seen in §4, adding indexed constraints indeed does lead to better capturing the dataset, as can be seen by the increasing log-likelihood as indexed constraints are added (as well as the trend for even higher log-likelihood as more sophisticated indexed constraint learners are used). However, as shown by the generalization experiment in §5, the indexation learners do not significantly impact the generalization ability of the resulting grammars, as the model with the best score on

the training data (iterative indexation) had a generalization score on par with the model that has no indexation. This is finding is similar to how adding random effects improves generalization in Mixed Effects models (e.g., Zymet 2018; Barr et al. 2013).

Of course, these results are based on one case study so far. While French schwa deletion is well-studied, there are many other cases of phonological generalization from lexically-conditioned processes (e.g., Zuraw 2000, Temkin-Martínez 2010, Linzen et al. 2013). Such cases should be investigated in the same framework in the future to obtain a fuller image of the potential of the currently proposed MaxEnt learners to generalize. Could it be that pre-training and post-training indexation work better than iterative indexation in certain cases? In addition, the iterative indexation learner itself should be analyzed in more detail. How can it be optimized to work for various cases? How conservative is it in terms of maximizing the phonological grammar's reluctance to take over phonological patterns with lexical patterns? These questions should be addressed in future work.

References

Barr, Dale J., Roger Levy, Cristoph Scheepers & Harry J. Tily. 2013. Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language* 68(3): 255–78.

Becker, Michael. 2009. *Phonological Trends in the Lexicon: The Role of Constraints.* Amherst, MA: University of Massachusetts Amherst dissertation.

Byrd, Richard H., Peihuang Lu, Jorge Nocedal & Ciyou Zhu. 1995. A Limited Memory Algorithm for Bound Constrained Optimization. *SIAM Journal on Scientific and Statistical Computing*, 16(5): 1190–208.

Côté, Marie-Hélène. 2000. *Consonant cluster phonotactics: a perceptual approach*. Cambridge, MA: Massachusetts Institute of Technology dissertation.

Dell, François. 1985. *Les règles et les sons: Introduction à la phonologie generative.* Paris: Hermann.

Goldwater, Sharon & Mark Johnson. 2003. Learning OT Constraint Rankings Using a Maximum Entropy Model. In Jennifer Spenader, Anders Eriksson & Östen Dahl (eds.), *Proceedings of the Stockholm Workshop on Variation within Optimality Theory*, 111–20. Stockholm: Department of Linguistics, Stockholm University.

Inkelas, Sharon & Cheryl Zoll. 2007. Is grammar dependence real? A comparison between cophonological and indexed constraint approaches to morphologically conditioned phonology. *Linguistics* 45(1): 133–71.

Kaplan, Aaron. 2011. Variation through Markedness Suppression. *Phonology* 28(3): 331–70.

Kraska-Szlenk, Iwona. 1995. *The Phonology of Stress in Polish*. Urbana-Champaign, IL: University of Illinois at Urbana-Champaign dissertation.

Kullback, Solomon & Richard Leibler. 1951. On Information and Sufficiency. *Annals of Mathematical Statistics* 22(1):79–86.

Linzen, Tal, Sofya Kasyanenko, and Maria Gouskova. 2013. Lexical and phonological variation in Russian prepositions. *Phonology* 30(3): 453-515.

Manning, Chris & Hinrich Schütze. 1999. Foundations of Statistical Natural Language Processing. Cambridge, MA: MIT Press.

Nazarov, Aleksei. 2018. Learning within- and between-word variation in probabilistic OT grammars. In Gillian Gallagher, Maria Gouskova, and Sora Heng Yin (eds.), *Supplemental Proceedings of the 2017 Annual Meeting on Phonology*, Washington, DC: LSA.

Nazarov, Aleksei. 2021. Learnability of indexed constraint analyses of phonological opacity. *Proceedings of the Society for Computation in Linguistics* 4, 16.

Pater, Joe. 2000. Non-uniformity in English secondary stress: The role of ranked and lexically specific constraints. *Phonology* 17(2): 237–74.

Pater, Joe. 2007. The Locus of Exceptionality: Morpheme-Specific Phonology as Constraint Indexation. In Leah Bateman, Michael O'Keefe, Ehren Reilly and Adam Werle (eds.), *University of Massachusetts Occasional Papers 32: Papers in Optimality Theory III*, 259–96. Amherst, MA: Graduate Linguistic Student Association.

Pater, Joe. 2010. Morpheme-specific phonology: Constraint indexation and inconsistency resolution. In Steve Parker (ed.), *Phonological argumentation: Essays on evidence and motivation*, 123–54. London: Equinox.

Racine, Isabelle. 2008. *Les effets de l'effacement du Schwa sur la production et la perception de la parole en français*. Geneva: Université de Genève dissertation.

Round, Erich. 2017. Phonological exceptionality is localized to phonological elements: The argument from learnability and Yidiny word-final deletion. In Claire Bowern, Larry Horn & Raffaella Zanuttini (eds.), *On looking into words (and beyond): Structures, relations, analyses*, 59–97. Berlin: Language Science Press.

Staubs, Robert. 2011. *Harmonic Grammar in R (hgR)*. Software package. http://blogs.umass.edu/hgr/

Temkin–Martínez, Michal. 2010. *Sources of Non-Conformity in Phonology: Variation and Exceptionality in Modern*

Hebrew Spirantization. Los Angeles, CA: University of Southern California dissertation.
Tesar, Bruce. 1995. *Computational Optimality Theory*. Boulder, CO: University of Colorado Boulder dissertation.
Zuraw, Kie. 2000. *Patterned Exceptions in Phonology*. Los Angeles, CA: University of California, Los Angeles dissertation.
Zymet, Jesse. 2018. *Lexical propensities in phonology: Corpus and experimental evidence, grammar, and learning.* Los Angeles, CA: University of California Los Angeles dissertation.