Chapter 1

Learning local transductions is hard

MARTIN JANSCHE

ABSTRACT. Local deterministic string-to-string transductions are generalizations of morphisms on free monoids. Learning local transductions reduces to inference of monoid morphisms. However, learning a restricted class of morphisms, the so-called fine morphisms, is an intractable problem, because the decision version of the empirical risk minimization problem contains an **NP**-complete subproblem.

1.1 Introduction

Symbolic approaches to natural language processing (NLP) based on finite automata (Roche and Schabes, 1997) suffer from a shortage of robust, practical inference procedures. If inductive inference is understood as 'identification in the limit' (Gold, 1967), then regular languages cannot be inferred on the basis of positive data alone. Most learning algorithms proposed for NLP tasks therefore employ different notions of inference, or aim at more restricted classes of languages, and they generally have to work with imperfect data.

This paper is about the problem of learning local transducers, a restricted subclass of the generalized sequential machines (Eilenberg, 1974), and inference is understood as empirical risk minimization. The general problem is illustrated by a specific NLP task, namely learning letter-to-sound rules (see for example van den Bosch, 1997) from a pronunciation dictionary. In this task, the training samples are pairs of strings, consisting of a string of letters – for example ⟨shoes⟩ – and a string of phonemes – for example /ʃuz/. Note that no relation between individual letters and phonemes is specified, which is to say one does **not** know whether the second symbol in /ʃuz/, the phoneme /u/, corresponds to the second symbol in ⟨shoes⟩, the letter ⟨h⟩. In this sense the present task is markedly different from other common NLP tasks – such as learning part-of-speech assignment rules – where explicit correspondences between input and output symbols exist.

Proceedings of Mathematics of Language 8 2003
R. T. Oehrle & J. Rogers (editors).

In general, a letter string may correspond to a longer phoneme string, for example[1]

⟨mutualism⟩ (9 letters)   /mjuʧəwəlɪzəm/ (12 phonemes),

or to a shorter phoneme string, such as

⟨featherweight⟩ (13 letters)   /fɛðɚwet/ (7 phonemes);

and even if the two strings happen to have the same length, as in

⟨parliamentarianism⟩ (18 letters)   /paɹləməntɛɹiənɪzəm/ (18 phonemes),

no alignment is implied. One usually assumes that letter strings are of equal length or longer than their corresponding phoneme strings. While clearly false in an absolute sense, this assumption is true for most English words (more than 98% of the entries in the CMU pronouncing dictionary), and workarounds for cases where it seems to break down have been suggested, for example the transcription system used by NETtalk (Sejnowski and Rosenberg, 1987).

Learning letter-to-sound rules can be conceptualized as grammatical inference of specific subclasses of rational transductions. For the class of subsequential transductions, limit-identification is possible (Oncina et al., 1993) and has been applied to the closely related problem of phonemic modeling (Gildea and Jurafsky, 1996), but only after modifications and incorporation of domain-specific knowledge. It can be shown that the algorithm proposed by Oncina et al. (1993) has poor out-of-class behavior and is brittle in the presence of imperfect data; furthermore its hypothesis space, the class of subsequential transductions, is arguably too general for the present task. Almost all approaches to learning letter-to-sound rules assume, justifiably, that the hypothesis space is restricted to the analog of the locally testable languages in the strict sense (McNaughton and Papert, 1972), which are limit-identifiable (García and Vidal, 1990). We call the analogous class of transductions *local transductions*.

Local transductions are computed by scanner transducers, which move a sliding window of fixed size across the input string and produce a string of output symbols for each window position; concatenating these output strings yields the overall output of the transducer. Since the size of the sliding window is fixed, one can assume without loss of generality that it is 1. If a larger window size n is needed, one can simply change the input alphabet to consist of n-tuples of symbols, and such a modified alphabet is obviously still finite; alternatively, one can think of

[1] The following examples are taken from the CMU pronouncing dictionary (Weide, 1998). Phonemic transcriptions have been changed to use IPA (International Phonetic Association, 1999).

this modification as a preprocessing step that applies a simple subsequential transducer to each input string. Functional transductions that examine individual input symbols (letters, or *n*-tuples of letters) without taking any context into account (a finite amount of history or lookahead can be incorporated into the modified symbols created by the preprocessing step) can be realized by generalized sequential machines with a trivial one-state topology and correspond exactly to morphisms on free monoids (Eilenberg, 1974, p. 299).

The subsequent discussion will refer to a finite set Σ of input symbols and a finite set Γ of output symbols. The free monoid generated by Σ is called Σ^* and has the property that every element (string) $x \in \Sigma^*$ has a unique factorization in terms of elements of Σ. This means that a morphism $g : \Sigma^* \to \Gamma^*$ on free monoids is completely characterized by $g|_\Sigma$, its restriction to Σ. Conversely, this allows us to define the following notion:

Definition 1.1 (Free monoid morphism). Given a function $f : \Sigma \to \Gamma^*$ define f^* to be the unique monoid morphism $f^* : \Sigma^* \to \Gamma^*$ such that $f^*(x) = f(x)$ for all $x \in \Sigma$; $f^*(\varepsilon) = \varepsilon$; and $f^*(yz) = f^*(y)\ f^*(z)$ for all $y, z \in \Sigma^*$.

At the core of the learning task is then the problem of finding a suitable function $f : \Sigma \to \Gamma^*$ mapping from individual input symbols to output strings. In this paper we focus on two classes of functions. The first class restricts the codomain to strings of length one. If $f : \Sigma \to \Gamma$ is such a function – an alphabetic substitution – then f^* is a *very fine morphism*, according to Eilenberg (1974, p. 6). The second class is a superset of the first and allows the empty string in the codomain. Eilenberg (1974) calls the morphism f^* a *fine morphism* if its underlying function f is of type $\Sigma \to \{\varepsilon\} \cup \Gamma$. For the specific problem of learning letter to sound rules we can restrict our attention to fine morphisms, since by our previous assumption letter strings are never shorter than their corresponding phoneme strings, so a fine morphism is formally adequate. In general we may want to consider other kinds of morphisms, for example those arising from functions of type $\Sigma \to \{\varepsilon\} \cup \Gamma \cup \Gamma^2$. However, most practically relevant classes of morphisms will probably contain the class of fine morphisms, and therefore many of their properties will carry over to more general settings. By restricting our attention to fine morphisms we have narrowed down the initial learning task considerably, as the hypothesis space H is now the set of functions of type $\Sigma \to \{\varepsilon\} \cup \Gamma$, which is always finite (though usually very large) for fixed finite alphabets Σ and Γ. Moreover, since $\ln |H| = |\Sigma|\ \ln(1 + |\Gamma|)$ the sample complexity of this hypothesis space is polynomial.

Our conceptualization of learning is not limit-identification, but empirical risk minimization. Although empirical risk minimization is somewhat problematic (Minka, 2000), particularly if the training data are not representative of the distribution of future data, it underlies most symbolic approaches to learning letter-to-sound rules, as well as many other NLP tasks. The empirical risk R of a hypothesis $h : \Sigma^* \to \Gamma^*$ is its average loss on a set of training samples $D \subseteq \Sigma^* \times \Gamma^*$, namely

$$R = \frac{1}{|D|} \sum_{\langle x,y\rangle \in D} L(h(x), y)$$

where $L : \Gamma^* \times \Gamma^* \to \mathbb{R}_{\geq 0}$ is the loss function. The most commonly used generally applicable loss functions for comparing strings are the zero-one loss

$$L_{\text{identity}}(y', y) = \begin{cases} 0 & \text{if } y' = y \\ 1 & \text{otherwise} \end{cases}$$

and $L_{\text{evenshtein}}$, the string edit distance (see for example Kruskal, 1983). Both kinds of loss play a role in the evaluation of letter-to-sound rules: for example, Damper et al. (1999, p. 164) use zero-one loss, and Fisher (1999) uses string edit distance. One generally requires that $L(y, y) = 0$, which is obviously the case for L_{identity}, and also holds for $L_{\text{evenshtein}}$ provided the cost for matching symbols is zero.

Empirical risk minimization under zero-one loss can mean one of two things: minimizing the total number of mistakes a hypothesis makes on the training data, or maximizing the number of correct predictions. These two notions are equivalent if optimal solutions can be found exactly, but differ for approximate solutions.[2]

1.2 Exact optimization

The problem of finding a function $f : \Sigma \to \{\varepsilon\} \cup \Gamma$ such that the empirical risk of f^* is minimal is fundamentally a combinatorial optimization problem. Like all such problems it can be stated formally in different ways (Papadimitriou and Steiglitz, 1998, p. 345f.): the optimization version asks for the optimal f for a given set of samples $D \subseteq \Sigma^* \times \Gamma^*$; the evaluation version asks for the total loss incurred on D by the optimal f^*; and the decision version asks whether there exists an f^* such that the total loss incurred by it on D is less than or equal to a given budget k. A solution to the optimization version could be used to construct an answer to the evaluation version, which in turn could be used to solve the decision version. Contrapositively, if the decision version is hard to solve, so are the other two versions.

Because $L(y, y) = 0$ for the loss functions L considered here, there is a common subproblem of the decision version which is independent of the loss function used:

[2] Suppose the true global optimum among 100 samples is 10 mistakes, and the optimum can be approximated within a ratio of 1.2. Approximate maximization would find a solution with at most $100 - 90/1.2 = 25$ mistakes, but approximate minimization yields a solution with at most $10 \times 1.2 = 12$ mistakes.

the restricted decision version asks whether there exists an f^* such that the total loss incurred by it on D is exactly zero. We call this the consistency problem. Obviously, if the decision version can be solved efficiently, so can the consistency problem.

Before we can formally state this key problem underlying the learning task, we need another auxiliary definition:

Definition 1.2 (Graph of a relation). Given a relation $R : A \to B$ on sets, define $\#R$, the *graph of R*, to be the set $\{\langle a, b\rangle \in (A \times B) \mid aRb\}$.

The consistency problems for the two classes of morphisms are stated in a format similar to the one used by Garey and Johnson (1979). An answer to the questions asked by these problem would tell us whether a suitable morphism exists that perfectly fits the training data D.

Problem 1.1 (Very Fine Morphism Consistency – VFMC)
Instance: A finite (multi)set $D \subseteq \Sigma^* \times \Gamma^*$ of training samples.

Question: Does there exist a very fine morphism consistent with all elements of D, i. e., is there a function $f : \Sigma \to \Gamma$ such that $D \subseteq \#(f^*)$?

Problem 1.2 (Fine Morphism Consistency – FMC)
Instance: A finite (multi)set $D \subseteq \Sigma^* \times \Gamma^*$ of training samples.

Question: Does there exist a fine morphism consistent with all elements of D, i. e., is there a function $f : \Sigma \to \{\varepsilon\} \cup \Gamma$ such that $D \subseteq \#(f^*)$?

The size of an instance of one of these problems is the total length of all strings in the training dictionary D:

Definition 1.3 (Dictionary size). Define the size $\|D\|$ of a dictionary $D \subseteq \Sigma^* \times \Gamma^*$ as

$$\|D\| = \sum_{\langle x,y\rangle \in D} |x| + |y|$$

where $|x|$ is the length of string x.

Of the two problems formulated here, FMC is intuitively more difficult than VFMC, since one has to decide which input symbols are mapped to the empty string, or equivalently, how the output strings should be aligned relative to the inputs. This issue does not arise with VFMC, since only strings of equal length need to be considered (if D contained a pair of strings with different lengths, then no very fine morphism can be consistent with D). It will be shown that FMC is a complete problem for the complexity class **NP** (see for example Garey and Johnson, 1979). Membership of FMC in **NP** can be established straightforwardly:

```
{Input: instance $D$, certificate $f$}
for all $\langle x, y\rangle \in D$ do
    $a_1 \cdots a_n \leftarrow x$
    $b_1 \cdots b_m \leftarrow y$
    $j \leftarrow 1$
    for $i \leftarrow 1$ to $n$ do
        if $f(a_i) \neq \varepsilon$ then
            if $j > m$ then
                return false
            else if $f(a_i) \neq b_j$ then
                return false
            else {$f(a_i)$ matches $b_j$}
                $j \leftarrow j + 1$
    if $j \neq m + 1$ then
        return false
return true
```

Figure 1.1: Certificate verification algorithm for FMC.

Theorem 1.1. *Problem* FMC *has succinct certificates that can be verified in polynomial time.*

Proof. A certificate for FMC is a partial function $f : \Sigma \to \Gamma \cup \{\varepsilon\}$, which can be represented in space linear in $\|D\|$ (because f only needs to mention elements of Σ that occur in D). Verification amounts to applying f^* to each input string in D and comparing the results to the corresponding reference output. The verification procedure, shown in Figure 1.1, runs in linear time and logarithmic space. □

As an aside, note that problem VFMC for very fine morphisms can be solved efficiently in linear time and space by the following procedure: iterate over D, and for each input symbol σ set $f(\sigma) \leftarrow \gamma$, where γ is the output symbol aligned[3] with σ; then run the verification algorithm from Figure 1.1 on D and f, and return its answer.

NP-hardness of FMC is established by a reduction from 3SAT, the decision problem asking whether there is a satisfying truth assignment for a set of disjunctive clauses with at most three literals each. We first define the construction and then prove that it correctly preserves the structure of 3SAT.

[3] A consistent very fine morphism can only exist if $|x| = |y|$ for all $\langle x, y\rangle \in D$, which means that x and y are aligned, in the sense that the nth symbol of x corresponds to the nth symbol of y.

Definition 1.4 (Boolean variable gadget). For any Boolean variable v, the set $\mathscr{V}(v)$ contains the following pairs (a_v and b_v are two new symbols dependent on v):

$$\langle a_v v \bar{v} b_v, FTF \rangle,$$
$$\langle a_v b_v, F \rangle.$$

Definition 1.5 (3SAT clause gadget). For any 3SAT clause C_i of the form $(l_{i1} \vee l_{i2} \vee l_{i3})$ (where each l_{ij} is a literal of the form v or $\bar{v}$) the set $\mathscr{C}(C_i)$ contains the following pairs (c_{ij}, d_{ij}, e_i and f_i for $1 \leq j \leq 3$ are eight new symbols dependent on i):

$$\langle c_{i1} l_{i1} d_{i1}, FT \rangle,$$
$$\langle c_{i2} l_{i2} d_{i2}, FT \rangle,$$
$$\langle c_{i3} l_{i3} d_{i3}, FT \rangle,$$
$$\langle d_{i1} d_{i2} d_{i3} e_i f_i, TT \rangle.$$

Definition 1.6 (Reduction from 3SAT). Given an instance $\varphi = \bigwedge_{i=1}^{n} C_i$ of 3SAT, define $\mathscr{D}(\varphi)$ as the collection $\bigcup_{i=1}^{n} \mathscr{C}(C_i) \cup \bigcup \{ \mathscr{V}(v) \mid \text{variable } v \text{ occurs in } \varphi \}$.

Theorem 1.2. *The reduction from* 3SAT *to* FMC *can be computed in logarithmic space and creates an instance whose size is polynomial in the size of the original instance.*

Proof. The reduction $\mathscr{D}$, which can be made to run in linear time, builds a collection $\mathscr{D}(\varphi)$ with the following properties: let m be the number of distinct variables of φ (so $m \leq 3n$); then $\|\mathscr{D}(\varphi)\| = 10m + 22n \leq 52n$, $|\mathscr{D}(\varphi)| = 2m + 4n \leq 10n$, $|\Sigma| = 4m + 8n \leq 20n$, and $|\Gamma| = 2$. Only counters need to be stored for computing the reduction (in order to keep track of clauses and variables represented by integers), which requires logarithmic space. □

Theorem 1.3. *Problem* FMC *is* **NP**-*hard.*

Proof. We show that $\varphi = \bigwedge_{i=1}^{n} C_i$ is satisfiable iff there exists a fine morphism f^* consistent with $\mathscr{D}(\varphi)$. It will be convenient to let V denote the set of distinct variables of φ.

($\Rightarrow$) Assume that φ is satisfiable, i. e., there exists a satisfying assignment $\tau : V \rightarrow \{T, F\}$. Incrementally define a fine morphism f^* consistent with $\mathscr{D}(\varphi)$ as follows: for all $v \in V$, let $f(v) = \tau(v)$ and $f(\bar{v}) = \overline{\tau(v)}$. If $\tau(v) = T$, let $f(a_v) = F$ and $f(b_v) = \varepsilon$, which makes f^* consistent with $\mathscr{V}(v)$; otherwise, if $\tau(v) = F$, let $f(a_v) = \varepsilon$ and $f(b_v) = F$ to make f^* consistent with $\mathscr{V}(v)$. In either case f^* can be made consistent with $\mathscr{V}(v)$, and because a_v and b_v do not occur outside the gadget for v, f^* can be made consistent with all variable gadgets.

The fact that τ is a satisfying assignment means that in each clause C_i at least one literal is made true by τ. So f will map at most two d_{ij} in $\mathscr{C}(C_i)$ to T, and therefore the definition of f^* can always be extended to make it consistent with the fourth pair in $\mathscr{C}(C_i)$ and hence consistent with the entire clause gadget for C_i. Since all symbols in a clause gadget other than literals of φ occur only in that gadget, the definition of f^* can be extended to make it consistent with all gadgets and therefore consistent with $\mathscr{D}(\varphi)$. Hence there exists a consistent fine morphism f^* constructible from τ.

($\Leftarrow$) Conversely, assume that a fine morphism g consistent with $\mathscr{D}(\varphi)$ exists. Show that $g|_V$, i. e. g restricted to the variables of φ, is a satisfying truth assignment for φ. The morphism g being consistent with $\mathscr{D}(\varphi)$ means that g is consistent with all variable gadgets and all clause gadgets.

Pick any variable gadget $\mathscr{V}(v)$. Then, because of the second pair in $\mathscr{V}(v)$, g must map exactly one of a_v and b_v to F: if $g(a_v) = F$ then $g(b_v) = \varepsilon$, and for the first pair $g(v) = T$ and $g(\overline{v}) = F$; otherwise if $g(b_v) = F$ then $g(a_v) = \varepsilon$, $g(v) = F$, and $g(\overline{v}) = T$. Note in particular that $(g|_V)(v) \in \{T, F\}$, so $g|_V$ is formally a truth assignment.

Now pick any clause gadget $\mathscr{C}(C_i)$ and suppose that g maps no l_{ij} in $\mathscr{C}(C_i)$ to T. Then all d_{ij} in $\mathscr{C}(C_i)$ are mapped to T because of the first three pairs in that clause gadget. But this would make g inconsistent with the fourth pair, contradicting the assumption that g is consistent with all clause gadgets. So g must map at least one l_{ij} in $\mathscr{C}(C_i)$ to T, which means that $g|_V$ makes the clause C_i true, and is therefore a satisfying truth assignment for φ. □

The preceding three theorems together imply that the consistency problem FMC is **NP**-complete. The existence of efficient algorithms for solving FMC is therefore unlikely. Since FMC is a subproblem of empirical risk minimization, the decision version of this optimization problem is also **NP**-complete.[4]

The evaluation and optimization version of empirical risk minimization do not seem to fall within the analogous class **FNP** of function problems. The reason for this is that an optimal solution f only certifies the existence of a feasible solution (namely f) within a certain budget k (namely the aggregate loss of f^* on the training data), but does not seem to provide enough information to verify in polynomial time that no better solution within a budget of $k-1$ can exist. It is doubtful whether

[4] Strictly speaking, the previous discussion only establishes **NP**-hardness of the decision version. Showing membership in **NP** is straightforward, but requires separate proofs depending on which loss function is used. For zero-one loss only a few minor modifications to the certificate verification algorithm in Figure 1.1 are required, which now has to aggregate the number of mistakes and compare it to the budget k. For loss based on edit distance, using the standard dynamic programming algorithm (Kruskal, 1983) ensures that certificates can be verified in polynomial time.

there are any polynomial-length certificates of optimality. We conjecture that these problems are in fact $\mathbf{FP^{NP}}$-complete, just like TSP (Papadimitriou, 1994).

1.3 Approximations and heuristics

Since the existence of exact efficient algorithms for solving the overall optimization problem is unlikely, one should consider the alternatives: approximate, heuristic, and/or inefficient algorithms.

Even for highly restricted problems the prospects are rather bleak. The optimization problem that maximizes empirical string-level classification accuracy (the dual of empirical zero-one loss, i. e. string-level classification error) for very fine morphisms will be called MAX-VFMC. It is far from clear whether MAX-VFMC is an easy or a hard problem, as we had shown earlier that VFMC can be solved very efficiently. We define the decision version of MAX-VFMC as follows:

Problem 1.3 (Very Fine Morphism Maximization – MAX-VFMC)
Instance: A finite sequence $D = \langle s_1, \ldots, s_n \rangle$ where each $s_i \in \bigcup_{j \in \mathbb{N}} \Sigma^j \times \Gamma^j$ for $1 \leq i \leq n$; and a natural number k with $k \leq n$.

Question: Does there exist a very fine morphism consistent with at least k elements of D, i. e., is there a function $f : \Sigma \to \Gamma$ and a length k unordered subsequence $\langle t_1, \ldots, t_k \rangle$ of D such that $t_i \in \#(f^*)$ for all $1 \leq i \leq k$?

We show that MAX-VFMC has probably (unless $\mathbf{P} = \mathbf{NP}$) no polynomial time approximation schemes (PTAS, which would allow us to find arbitrarily good approximations efficiently). In the best case, there may be an approximation algorithm for MAX-VFMC with a fixed approximation ratio, which would make MAX-VFMC a member of the class **APX** (Ausiello et al., 1999); whether or not this is the case is an open question.

Theorem 1.4. *Problem* MAX-VFMC *is* **APX***-hard.*

Proof. Show this by exhibiting an **AP**-reduction from an **APX**-complete problem. It suffices to show that MAX-k-CSP is L-reducible (Papadimitriou, 1994, 309ff.) to MAX-VFMC. MAX-k-CSP is a constraint satisfaction problem (Khanna et al., 1997) with conjunctive constraints containing at most k literals (see also Ausiello et al., 1999).

Given an instance $\varphi = \langle (l_{11} \wedge \cdots \wedge l_{1k}), \ldots, (l_{n1} \wedge \cdots \wedge l_{nk}) \rangle$ of MAX-k-CSP, construct an instance of MAX-VFMC by mapping the ith constraint $(l_{i1} \wedge \cdots \wedge l_{ik})$ to the pair $\langle l_{i1}\overline{l_{i1}} \ldots l_{ik}\overline{l_{ik}}, TF \ldots TF \rangle$ to form D (if a literal l is negative, i. e. of the form $\bar{v}$, then $\bar{l}$ is simply v). So Σ consist of the negated and unnegated variables of φ, and $\Gamma = \{T, F\}$. This construction ensures that there is a truth assignment

τ that makes exactly m constraints of φ true iff there exists a very fine morphism f^* which is consistent with exactly m elements of D. One can construct f from τ (and vice versa) via $f(v) = \tau(v)$ and $f(\overline{v}) = \overline{\tau(v)}$ where v is a variable occurring in φ. □

Exact global optimization of MAX-VFMC is theoretically possible via branch-and-bound search. While this inefficient algorithm can be used for very small problem instances (learning English letter-to-sound rules with no conditioning context, for which only a few trillion morphisms have to be explored), it becomes intractable for even slightly larger problems (for English letter-to-sound rules conditioned on one letter of context there are more than one trequadragintillion feasible solutions). Heuristic algorithms, especially those based on local search (Papadimitriou and Steiglitz, 1998), are efficient and do in practice improve on greedily constructed initial solutions, but offer no performance guarantees.

1.4 Conclusions

We have reduced the problem of learning local transductions to the problem of learning morphisms on free monoids (the reduction may involve deterministic preprocessing of the training data). The restricted problem of deciding whether there exists a fine morphism consistent with a set of training samples was shown to be **NP**-complete. Since this problem is a specialization of the decision version of empirical risk minimization under any loss function L for which $L(y,y) = 0$, the larger optimization problems which generalize the consistency problem are generally intractable.

Acknowledgements

This paper is based, occasionally verbatim, on the author's dissertation (Jansche, to appear 2003), which contains further details. Many thanks to Chris Brew, Gerald Penn, and Richard Sproat, as well as two anonymous reviewers, for comments and feedback. The usual disclaimers apply.

Bibliography

Ausiello, G., P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi (1999). *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties.* Springer, Berlin.

Damper, R. I., Y. Marchand, M. J. Adamson, and K. Gustafson (1999). Evaluating the pronunciation component of text-to-speech systems for English: A performance comparison of different approaches. *Computer Speech and Language*, **13**(2):155–176.

Eilenberg, S. (1974). *Automata, Languages, and Machines*, volume A. Academic Press, New York.

Fisher, W. M. (1999). A statistical text-to-phone function using ngrams and rules. In *International Conference on Acoustics, Speech, and Signal Processing*, pp. 649–652. Phoenix, AZ.

García, P. and E. Vidal (1990). Inference of *k*-testable languages in the strict sense and application to syntactic pattern recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **12**(9):920–925.

Garey, M. R. and D. S. Johnson (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, San Francisco.

Gildea, D. and D. Jurafsky (1996). Learning bias and phonological-rule induction. *Computational Linguistics*, **22**(4):497–530.

Gold, E. M. (1967). Language identification in the limit. *Information and Control*, **10**(5):447–474.

International Phonetic Association (1999). *Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet*. Cambridge University Press, Cambridge, England.

Jansche, M. (to appear 2003). *Inference of String Mappings*. Ph.D. thesis, The Ohio State University, Columbus, OH.

Khanna, S., M. Sudan, and D. P. Williamson (1997). A complete classification of the approximability of maximization problems derived from Boolean constraint satisfaction. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pp. 11–20. El Paso, TX.

Kruskal, J. B. (1983). An overview of sequence comparison. In D. Sankoff and J. Kruskal, eds., *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, pp. 1–44. Addison-Wesley, Reading, MA. Reissued by CSLI Publications, Stanford, CA, 1999.

McNaughton, R. and S. Papert (1972). *Counter-Free Automata*. MIT Press, Cambridge, MA.

Minka, T. P. (2000). Empirical risk minimization is an incomplete inductive principle. `http://www.stat.cmu.edu/˜minka/papers/erm.html`.

Oncina, J., P. García, and E. Vidal (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, **15**(5):448–458.

Papadimitriou, C. H. (1994). *Computational Complexity*. Addison-Wesley, Reading, MA.

Papadimitriou, C. H. and K. Steiglitz (1998). *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, Mineola, NY. Originally published by Prentice Hall, Englewood Cliffs, NJ, 1982.

Roche, E. and Y. Schabes, eds. (1997). *Finite-State Language Processing*. Language, Speech and Communication. MIT Press, Cambridge, MA.

Sejnowski, T. J. and C. R. Rosenberg (1987). Parallel networks that learn to pronounce English text. *Complex Systems*, **1**(1):145–168.

van den Bosch, A. P. J. (1997). *Learning to Pronounce Written Words: A Study in Inductive Language Learning*. Ph.D. thesis, Universiteit Maastricht, Maastricht, The Netherlands.

Weide, R. L. (1998). The Carnegie Mellon pronouncing dictionary version 0.6. Electronic document, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. `ftp://ftp.cs.cmu.edu/project/fgdata/dict/`.