Probabilistic Finite State Machines

Using and building

Franck Thollard

http://eurise.univ-st-etienne.fr/~thollard

EURISE
Saint-Étienne
France
Contents

- Probabilistic Finite State Machines
Contents

- Probabilistic Finite State Machines
- What learning means
Contents

• Probabilistic Finite State Machines
• What learning means
• Learnability results
Contents

- Probabilistic Finite State Machines
- What learning means
- Learnability results
- Algorithmic issues
Contents

• Probabilistic Finite State Machines
• What learning means
• Learnability results
• Algorithmic issues
• Experimental issues
Contents

- Probabilistic Finite State Machines
  - n-grams / MM / PST / A-PDFA
  - PDFA / Residual automata
  - Other : PFA / HMM, PCFG, ...
  - Choosing a model

- What learning means
- Learnability results
- Algorithmic issues
- Conclusions
The probability of a sequence

Computation using the chain rule:

\[
P(he \text{ reads a book}) = P(he) \times P(\text{reads}|he) \\
\times P(a|he \text{ reads}) \times P(\text{book}|he \text{ reads a})
\]
The probability of a sequence

Computation using the chain rule:

\[ P(\text{he reads a book}) = P(\text{he}) \times P(\text{reads}|\text{he}) \times P(\text{a}|\text{he reads}) \times P(\text{book}|\text{he reads a}) \]

and more generally:

\[ P(w_1w_2\ldots w_n) = P(w_1) \times P(w_2|w_1) \times \ldots \times P(w_n|w_1w_2\ldots w_{n-1}) \]
The probability of a sequence

Computation using the chain rule:

\[
P(he \text{ reads a book}) = P(he) \times P(\text{reads}|he) \times P(a|he \text{ reads}) \times P(\text{book}|he \text{ reads a})
\]

and more generally:

\[
P(w_1w_2\ldots w_n) = P(w_1) \times P(w_2|w_1) \times \ldots \times P(w_n|w_1w_2\ldots w_{n-1})
\]

Definition: \(w_1w_2\ldots w_{n-1}\) is called the history.
Some models

See (Vidal et al., 2005) for a survey

• $n$-grams / MM $\Leftrightarrow k$-testables automata
Some models

See (Vidal et al., 2005) for a survey

- \( n \)-grams / MM ⇔ \( k \)-testables automata
- Probabilistic Automata without cycles
Some models

See (Vidal et al., 2005) for a survey

- $n$-grams / MM $\Leftrightarrow k$-testables automata
- Probabilistic Automata without cycles
- Probabilistic Deterministic Automata

Note: models define a pdf on $n$, for each $n$ add of eos symbol)
Some models

See (Vidal et al., 2005) for a survey

- $n$-grams / MM $\Leftrightarrow$ $k$-testables automata
- Probabilistic Automata without cycles
- Probabilistic Deterministic Automata
- Residual Automata (Esposito & al., 2002)
Some models

See (Vidal et al., 2005) for a survey

- $n$-grams / MM $\Leftrightarrow k$-testables automata
- Probabilistic Automata without cycles
- Probabilistic Deterministic Automata
- Residual Automata (Esposito & al., 2002)
- Probabilistic non-deterministic Automata ($\Leftrightarrow$ HMM)
Some models

See (Vidal et al., 2005) for a survey

- $n$-grams / MM $\Leftrightarrow k$-testables automata
- Probabilistic Automata without cycles
- Probabilistic Deterministic Automata
- Residual Automata (Esposito & al., 2002)
- Probabilistic non-deterministic Automata ($\Leftrightarrow$ HMM)
- PCFG: Probabilistic Context-Free Grammars
Some models

See (Vidal et al., 2005) for a survey

- $n$-grams / MM $\Leftrightarrow k$-testables automata
- Probabilistic Automata without cycles
- Probabilistic Deterministic Automata
- Residual Automata (Esposito & al., 2002)
- Probabilistic non-deterministic Automata ($\Leftrightarrow$ HMM)
- PCFG: Probabilistic Context-Free Grammars

Note: models define a pdf on $\Sigma^n$, for each $n$

add of $\text{eos}$ symbol $\Rightarrow$ pdf on $\Sigma^*$
The $n$-grams model (1/3)

**Assumption:** the history is supposed bound

Example: history of size one

2-grams (known as bigram)

$$P(he \ reads \ a \ book) = P(he) \times P(reads | he) \times P(a | reads) \times P(book | a)$$
The $n$-grams model (1/3)

**Assumption:** the history is supposed bound

Example: history of size one $\Rightarrow$ 2-grams (known as bigram)
The \( n \)-grams model (1/3)

**Assumption:** the history is supposed bound

Example: history of size one \( \Rightarrow \) 2-grams (known as bigram)

\[
P(\text{he reads a book}) = P(\text{he}) \times P(\text{reads}|\text{he}) \times P(\text{a}|\text{reads}) \times P(\text{book}|\text{a})
\]
The \( n \)-grams model (2/3)

Estimating \( n \)-grams probabilities

The probabilities are estimated using a corpus by counting occurrences of the \( n \)-uplets:

\[
P(book|a) = \frac{Ct(a \ book)}{Ct(a)}
\]
The $n$-grams (3/3)

Automata representation of $n$-grams

The $n$-grams (3/3)

Automata representation of $n$-grams

The automaton represents the sequence of words with their probabilities:

- State 0 with the word "the" (0.9)
- State 1 with the word "student" (0.4)
- State 2 with the word "eating" (0.4) and "a" (0.4)
- State 3 with the word "mans" (0.6) and "like" (0.6)
- State 4 with the word "book" (1)
- State 5
- State 6
- State 7 with the word "empanadas" (1)

Probabilistic Finite State Machines – p.8/38
Cyclic Automaton

unbounded history

A Probabilistic automaton

Note: $P(papers | \ldots)$ depends on an unbound history.
## Choosing a model

<table>
<thead>
<tr>
<th>Model</th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n)-gram</td>
<td>Easy to build, Good estimates</td>
<td>Assumption can be false, Big model</td>
</tr>
</tbody>
</table>

Probabilistic Finite State Machines – p.10/38
## Choosing a model

<table>
<thead>
<tr>
<th>Model</th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$-gram</td>
<td>Easy to build, Good estimates</td>
<td>Assumption can be false Big model</td>
</tr>
<tr>
<td>A-PDFA</td>
<td>History larger</td>
<td>Building the structure</td>
</tr>
</tbody>
</table>
# Choosing a model

<table>
<thead>
<tr>
<th>Model</th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$-gram</td>
<td>Easy to build, Good estimates</td>
<td>Assumption can be false</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Big model</td>
</tr>
<tr>
<td>A-PDFA</td>
<td>History larger</td>
<td>Building the structure</td>
</tr>
<tr>
<td>PDFA</td>
<td>Unbound history</td>
<td>Building the structure</td>
</tr>
<tr>
<td></td>
<td>Parsing time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Correct results</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Size of the model</td>
<td></td>
</tr>
</tbody>
</table>
## Choosing a model

<table>
<thead>
<tr>
<th>Model</th>
<th>Good</th>
<th>Bad</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$-gram</td>
<td>Easy to build, Good estimates</td>
<td>Assumption can be false Big model</td>
</tr>
<tr>
<td>A-PDFA</td>
<td>History larger</td>
<td>Building the structure</td>
</tr>
<tr>
<td>PDFA</td>
<td>Unbound history</td>
<td>Building the structure</td>
</tr>
<tr>
<td></td>
<td>Parsing time</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Correct results</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Size of the model</td>
<td></td>
</tr>
<tr>
<td>PFA / HMM</td>
<td>Unbound history</td>
<td>Parsing time or &quot;Loose&quot; of proba in non determinism</td>
</tr>
<tr>
<td></td>
<td>Hand made structure</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Size of the model</td>
<td></td>
</tr>
</tbody>
</table>
Contents

- Probabilistic Finite State Machines
- What learning means
  - What learning means?
  - What can be learned?
- Learnability results
- Algorithmic issues
- Experimental issues
- Conclusions
Machine Learning Assumption

Formal Language Learning

aabb → INERENCE → FORMAL LANGUAGE

aabb
aaabb
aaaaabbb
What learning means?

$\mathcal{D}$

$\mathcal{C}$

Generation($\mathcal{D}, \mathcal{C}$)

Inference

$D_1$

$D_2$

$\ldots D_n$

$G_1$

$G_2$

$\ldots G_n$
What learning means?

![Diagram showing the process of learning with symbols D, C, G, D\_1, D\_2, ..., D\_n, G\_1, G\_2, ..., G\_n.]
What learning means?

Formalization

- Learning criterion
What learning means?

Formalization

- Learning criterion

- Learnability results (w.r.t. automata)
What learning means?

Formalization

- Learning criterion
  - Identification in the limit
  - PAC learning
Identification in the limit
PAC Learning

Proba–D–PAC

Generation(D,C) → Inference

D → D₁, D₂, ..., Dₙ

G → G₁, G₂, ..., Gₙ

??
PAC Learning

Proba–D–PAC

\( \mathcal{D} \)

\( \mathcal{C} \)

Generation(\( \mathcal{D}, \mathcal{C} \))

Inference

\( D_1 \)

\( D_2 \)

\( \ldots \)

\( D_n \)

\( G_1 \)

\( G_2 \)

\( \ldots \)

\( G_n \)

\( Pr \)
PAC Learning

Proba–D–PAC

\[
\begin{align*}
D & \xrightarrow{D} \text{Generation}(D,C) \\
C & \xrightarrow{G} \text{Inference} \\
\text{Generation}(D,C) & \rightarrow D_1, D_2, \ldots, D_n \\
\text{Inference} & \rightarrow G_1, G_2, \ldots, G_n \\
\Pr[D(G, G_n) < \epsilon] &
\end{align*}
\]
PAC Learning

Proba–D–PAC

\[ Pr[D(G, G_n) < \epsilon] > \delta \]
PAC Learning

Proba–D–PAC

Link between

Precision / Confidence / number of examples / complexity

of the class / . . .
Identification in the limit
(With Proba One)

- The class of recursively enumerable languages can be identified in the limit with probability one (Horning, 1969).
Identification in the limit
(With Proba One)

- The class of recursively enumerable languages can be identified in the limit with probability one (Horning, 1969).
- The class of probabilistic automata can be identified in the limit with probability one (constructive proof) (Carrasco, 1999).
Proba-Poly-PAC (1/2)

Possible

- Proba-$d_\infty$-PAC PFA (Angluin 88, lemma 14)
Proba-Poly-PAC (1/2)

Possible

- Proba-$d_\infty$-PAC PFA (Angluin 88, lemma 14)
- Proba-KL-PAC Unigram with unknown vocabulary (Mc Allester & Shapire, 2000)
Proba-Poly-PAC (1/2)

Possible

- Proba-\(d_\infty\)-PAC PFA (Angluin 88, lemma 14)
- Proba-KL-PAC Unigram with unknown vocabulary (Mc Allester & Shapire, 2000)
- Proba-KL-PAC PFA on \(\Sigma^n, \Sigma\) and \(n\) known, nb of states known (Abe & Warmuth, 1992)
Proba-Poly-PAC (1/2)

Possible

- Proba-$d_\infty$-PAC PFA (Angluin 88, lemma 14)
- Proba-KL-PAC Unigram with unknown vocabulary (Mc Allester & Shapire, 2000)
- Proba-KL-PAC PFA on $\Sigma^n$, $\Sigma$ and $n$ known, nb of states known (Abe & Warmuth, 1992)
- Proba-KL-PAC Acyclic PDFA on $\Sigma^n$, nb States known, $\Sigma$ and $n$ known (Ron & al., 1995)
Proba-Poly-PAC (1/2)

Possible

- Proba-$d_\infty$-PAC PFA (Angluin 88, lemma 14)
- Proba-KL-PAC Unigram with unknown vocabulary (McAllester & Shapire, 2000)
- Proba-KL-PAC PFA on $\Sigma^n$, $\Sigma$ and $n$ known, nb of states known (Abe & Warmuth, 1992)
- Proba-KL-PAC Acyclic PDFA on $\Sigma^n$, nb States known, $\Sigma$ and $n$ known (Ron & al., 1995)
- Proba-KL-PAC PDFA Cyclic Aut structure (Thollard & Clark, 2004)
Proba-Poly-PAC (1/2)

Possible

• Proba-$d_\infty$-PAC PFA (Angluin 88, lemma 14)
• Proba-KL-PAC Unigram with unknown vocabulary (Mc Allester & Shapire, 2000)
• Proba-KL-PAC PFA on $\Sigma^n$, $\Sigma$ and $n$ known, nb of states known (Abe & Warmuth, 1992)
• Proba-KL-PAC Acyclic PDFA on $\Sigma^n$, nb States known, $\Sigma$ and $n$ known (Ron & al., 1995)
• Proba-KL-PAC PFDA Cyclic Aut structure (Thollard & Clark, 2004)
• Proba-KL-PAC PDFA Cyclic Aut + informations (Clark & Thollard, 2004)
Proba-Poly-PAC (2/2)

Impossible

- Proba AFN on $\Sigma^n$, $\Sigma$ unknown (Abe & Warmuth, 1992)
- PDF of unknown class on $\{0, 1\}^n$ (Kearns & al., 1994)
- Proba-KL-PAC Cyclic Aut, without aut information (Clark & Thollard, 2002)
Contents

• Probabilistic Finite State Machines
• What learning means
• Learnability results
• Algorithmic issues
  • Template technique
  • Instantiation of the template technique
  • Smoothing automata
• Experimental issues
• Conclusions
Template algorithm

The common strategy follows two steps

- Building a maximum likelihood estimate of the data
Template algorithm

The common strategy follows two steps

- Building a maximum likelihood estimate of the data
- Generalizing using state merging operations.
Learning by heart: the PPTA

PPTA of the learning set
\[ EA = \{ \lambda, aac, aaa, aac, abc, aac, abc, \lambda, a, ab \} \]
Learning by heart: the PPTA

PPTA of the learning set
\[ EA = \{ \lambda, aac, aaa, aac, abc, aac, abc, abc, \lambda, a, ab \} \]
Learning by heart: the PPTA

PPTA of the learning set
EA = {λ, aac, aaa, aac, abc, aac, abc, abc, λ, a, ab }

Note: String "aba" has null probability
Generalization

Choosing two states

0 (2/11) \rightarrow 1 (1/9) \rightarrow 2 (1/4) \rightarrow 4 (3/3)

a_9/11 \rightarrow b (4/9) \rightarrow c (3/4)

3 (0/4) \rightarrow 5 (3/3)

a_4/9 \rightarrow c_3/4

3 (1/1)

a (1/4)

Probabilistic Finite State Machines – p.23/38
Generalization

Choosing two states

Choosing state 2 and 3
Generalization

Merging states 2 and 3: can lead to non determinism
Generalization

Merging states 4 and 5

Merging states 4 and 5
Generalization

After the "determinization"

Note: String "aba" has non null probability
Generic Algorithm

Input: A multiset of string EA, a real $\alpha$
Output: A PDFA

begin
  Building-PPTA (EA);
  $A \leftarrow PPTA$;
  while $(q_i, q_j) \leftarrow Choosing-Two-States (A)$ do
    if Compatible $(i, j, \alpha)$ then
      $A \leftarrow Merge (A, q_i, q_j)$;
    end
  end
end
Return $A$;
end
Ordering merges

PPTA built on the multiset

$$EA = \{ \lambda, aac, aac, abc, aac, aac, abc, abc, \lambda, a, ab \}$$
Merging ordering

Alergia (Carrasco & Oncina, 1994):

HMM-infer (Stolcke, 1994): looking at each merge at each time → not tracktable on big data sets

LAPPTA (Ron & al., 1995): building of acyclic automata

EDSM (Lang, 1998): merging ordering based on the quantity of information

DDSM (Thollard, 2001): ordering adapted from the EDSM algorithm.
Compatibility tests

Alergia (Carrasco, 1994): Statistic test based on Hoeffding bounds

LAPPTA (Ron & al., 1995): Statistic test based on similarity measure

Youg-Lai & Tompa, (2000): Same as Alergia but emphasis on low frequency problem

MDI (Thollard & al., 2000): Tradeoff between size and distance to the data

M-Alergia (Kermorvant & Dupont, 2002): Statistical test based on multinomial test

Alergia (Habrard & al., 2003): Defines and deal with uniform noise.
Other learning schemes

- Splitting/merging strategy (Brant, 1996)
The smoothing problem

The farm example

Let F be a farm with:

- 3 chickens
- 2 ducks

What is the probability of:
The smoothing problem

The farm example

Let F be a farm with:

- 3 chickens
- 2 ducks

What is the probability of:

Pr(chicken) = 3/5
The smoothing problem

The farm example

Let F be a farm with:

- 3 chickens
- 2 ducks

What is the probability of:

$Pr(\text{pig}) = 0$
The smoothing problem

The farm example

Let $F$ be a farm with:

- 3 chickens
- 2 ducks

What is the probability of:

$Pr(\text{lion}) = 0$
The smoothing problem

Considering the chain rule

\[ n \text{-grams} \] the problem needs to be considered **only** with null probability \( n \text{-grams} \).

Automata

- estimating the first null transition ?
- where to continue in the automaton ? in the same automaton ? on others ?
The smoothing problem

Probabilistic Finite State Machines – p.31/38
Smoothing automata

LAPPTA (Ron & al., 1995): Creation of a "small frequency state"

Alergia2 (Young-lai & Tompa, 2000): Emphasis on small frequency transitions

Error-correcting (Dupont & Amengual, 2000): Error correcting

Discounting (Thollard, 2001): Back-off to a unigram

Discounting (Llorens & al., 2002): Back-off to automata

Additive discounting (Thollard & Clark, 2004): Theoretical justification.

Discounting (Mc Allester & Shapire, 2000): Theoretical discounting for the unigram.
Contents

• Probabilistic Finite State Machines
• What learning means
• Learnability results
• Algorithmic issues
• Experimental issues
  • Experimentation with MDI/DDSM
  • Around the inference
• Conclusions
Experimental issues (MDI/DDSM)

- **Language Modeling:** much compact models (Thollard, 2001),
Experimental issues (MDI/DDSM)

- **Language Modeling:** much compact models (Thollard, 2001),

- **Speech to text:** faster at parsing time than bigrams (Experimentation at CNET),
Experimental issues (MDI/DDSM)

- **Language Modeling**: much compact models (Thollard, 2001),
- **Speech to text**: faster at parsing time than bigrams (Exeperimentation at CNET),
- **Noun phrase chunking**: competitive results (~ 90 %) (Thollard & Clark, 2004),
Experimental issues (MDI/DDSM)

- **Language Modeling**: much compact models (Thollard, 2001),

- **Speech to text**: faster at parsing time than bigrams (Exeperimentation at CNET),

- **Noun phrase chunking**: competitive results (~ 90 %) (Thollard & Clark, 2004),

- **Body rule generation**: better and more compact than $n$-gram (Infante-Lopez, 2004).
Around the inference

Clustering (Dupont & Chase, 1998)
Interpolating automata (Thollard, 2001)
Bagging (Thollard & Clark, 2002)
Boosting (Thollard & al. 2002)
Typing automata (Kermorvant & de la Higuera, 2002)
Contents

• Probabilistic Finite State Machines
• What learning means
• Learnability results
• Algorithmic issues
• Experimental issues
• Conclusions
Conclusion

Probabilistic grammatical inference

- Framework in which theoretical results exist
- Good results in many domains (e.g. NLP)
- Can deal with big data sets (e.g. Wall Street Journal)
- Provides very compact automata.
Open questions

Theoretical:

- Learning/smoothing $n$-gram models
- What is a good distance for the Proba-D-PAC framework?

Practical:

- Learning non-deterministic models
- Improving the merging ordering
- Algorithmic: improving the algorithms