Maximum Entropy and Language Processing

Georg Holzmann

7. Dezember 2006

An Example

Let's introduce the concept of maximum entropy through a simple example:

- want to model a proper French translation of the English word *in*
- we collect a lot of examples from expert translators (*feature selection*)
- and then try to construct a model of this process (*model selection*)

An Example

Let's introduce the concept of maximum entropy through a simple example:

- want to model a proper French translation of the English word *in*
- we collect a lot of examples from expert translators (*feature selection*)
- and then try to construct a model of this process (*model selection*)

An Example

Let's introduce the concept of maximum entropy through a simple example:

- want to model a proper French translation of the English word *in*
- we collect a lot of examples from expert translators (*feature selection*)
- and then try to construct a model of this process (*model selection*)

An Example

- first observation: expert tanslator always chooses among these five French phrases:

  dans, en, à, au cours de, pendant

- so we can define the first constraint on our model p:

$$p(dans)+p(en)+p(à)+p(au\ cours\ de)+p(pendant) = 1$$

- with only this knowledge, the most appealing model is the uniform model:

$$p(dans)=1/5;\ p(en)=1/5;\ p(à)=1/5;$$
$$p(au\ cours\ de)=1/5;\ p(pendant)=1/5;$$

An Example

- first observation: expert tanslator always chooses among these five French phrases:

  dans, en, à, au cours de, pendant

- so we can define the first constraint on our model p:

  $$p(dans)+p(en)+p(à)+p(au\ cours\ de)+p(pendant) = 1$$

- with only this knowledge, the most appealing model is the uniform model:

  $$p(dans)=1/5;\ p(en)=1/5;\ p(à)=1/5;$$
  $$p(au\ cours\ de)=1/5;\ p(pendant)=1/5;$$

An Example

- first observation: expert tanslator always chooses among these five French phrases:

  dans, en, à, au cours de, pendant

- so we can define the first constraint on our model p:

  p(dans)+p(en)+p(à)+p(au cours de)+p(pendant) = 1

- with only this knowledge, the most appealing model is the uniform model:

  p(dans)=1/5; p(en)=1/5; p(à)=1/5;
  p(au cours de)=1/5; p(pendant)=1/5;

An Example

- one more observations: the translator chose either *dans* or *en* in 30% of the time:

$$p(dans)+p(en) = 3/10$$

- our new model is again the most uniform:

$$p(dans)=3/20;\ p(en)=3/20;\ p(à)=7/30;$$
$$p(au\ cours\ de)=7/30;\ p(pendant)=7/30;$$

- on more contraint: $p(dans)+p(à) = 1/2$:
now the calculation is not that easy anymore ...

An Example

- one more observations: the translator chose either *dans* or *en* in 30% of the time:

$$p(dans){+}p(en) = 3/10$$

- our new model is again the most uniform:

$$p(dans){=}3/20;\ p(en){=}3/20;\ p(\grave{a}){=}7/30;$$
$$p(au\ cours\ de){=}7/30;\ p(pendant){=}7/30;$$

- on more contraint: $p(dans){+}p(\grave{a}) = 1/2$:
  now the calculation is not that easy anymore ...

An Example

- one more observations: the translator chose either *dans* or *en* in 30% of the time:

$$p(dans)+p(en) = 3/10$$

- our new model is again the most uniform:

$$p(dans)=3/20;\ p(en)=3/20;\ p(à)=7/30;$$
$$p(au\ cours\ de)=7/30;\ p(pendant)=7/30;$$

- on more contraint: $p(dans)+p(à) = 1/2$:
now the calculation is not that easy anymore ...

The Maximum Entropy Principle

- with the last constraint we introduced two problems:
  1. What exactly is meant by "most uniform"?
  2. How to calculate the model according to those constraints?
- the maximum entropy method (ME) tries to answer both these questions
- the ME-principle is simple:

  model all that is known and assume nothing about that which is unknown

The Maximum Entropy Principle

- with the last constraint we introduced two problems:
  1. What exactly is meant by "most uniform"?
  2. How to calculate the model according to those constraints?
- the maximum entropy method (ME) tries to answer both these questions
- the ME-principle is simple:

  model all that is known and assume nothing about that which is unknown

The Maximum Entropy Principle

- with the last constraint we introduced two problems:
  1. What exactly is meant by "most uniform"?
  2. How to calculate the model according to those constraints?
- the maximum entropy method (ME) tries to answer both these questions
- the ME-principle is simple:

model all that is known and assume nothing about that which is unknown

The Maximum Entropy Principle

- with the last constraint we introduced two problems:
  1. What exactly is meant by "most uniform"?
  2. How to calculate the model according to those constraints?
- the maximum entropy method (ME) tries to answer both these questions
- the ME-principle is simple:

model all that is known and assume nothing about that which is unknown

The Maximum Entropy Principle

- with the last constraint we introduced two problems:
  1. What exactly is meant by "most uniform"?
  2. How to calculate the model according to those constraints?
- the maximum entropy method (ME) tries to answer both these questions
- the ME-principle is simple:

  model all that is known and assume nothing about that which is unknown

Definition of the Model

the model can be considered as a random process with the following properties:

- produces an output value y, which is a member of a finite set Y
  (in the previous example y was one word of the set *dans, en, à, au cours de, pendant*)
- the model is influenced by some contextual information x, a meber of a finite set X
  (the english word *in* in the previous example)
- the model is the conditional probability that, given a context x, the process will output y
  we will notate it as $p(y \mid x)$, which is a member of the set of all conditional probability distributions P

Definition of the Model

the model can be considered as a random process with the following properties:

- produces an output value y, which is a member of a finite set Y
  (in the previous example y was one word of the set *dans, en, à, au cours de, pendant*)
- the model is influenced by some contextual information x, a meber of a finite set X
  (the english word *in* in the previous example)
- the model is the conditional probability that, given a context x, the process will output y
  we will notate it as $p(y \mid x)$, which is a member of the set of all conditional probability distributions P

Definition of the Model

the model can be considered as a random process with the following properties:

- produces an output value y, which is a member of a finite set Y
  (in the previous example y was one word of the set *dans, en, à, au cours de, pendant*)
- the model is influenced by some contextual information x, a meber of a finite set X
  (the english word *in* in the previous example)
- the model is the conditional probability that, given a context x, the process will output y
  we will notate it as $p(y \mid x)$, which is a member of the set of all conditional probability distributions P

Training Data

- a large number of samples $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ are taken e.g. from an expert translator
- we can create the empirical probability distribution $\tilde{p}$ of the training data:

$$\tilde{p}(x, y) = \frac{1}{N} \times \textit{nmber of times that (x,y) occurs}$$

Training Data

- a large number of samples $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ are taken e.g. from an expert translator
- we can create the empirical probability distribution $\tilde{p}$ of the training data:

$$\tilde{p}(x, y) = \frac{1}{N} \times \textit{nmber of times that (x,y) occurs}$$

Feature Function

For each constraint we know, we create a so called feature function:

- For instance, if in the training data *April* is the word following *in*, the translation of *in* is *en* in 90%
- we express the feature in a indicator function:

$$f(x,y) = \begin{cases} 1 & \text{if x = en and April follows in} \\ 0 & \text{otherwise} \end{cases}$$

- so we can calculate the expected value of that feature:

$$\tilde{p}(f) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Feature Function

For each constraint we know, we create a so called feature function:

- For instance, if in the training data *April* is the word following *in*, the translation of *in* is *en* in 90%
- we express the feature in a indicator function:

$$f(x,y) = \begin{cases} 1 & \text{if x} = \text{en and April follows in} \\ 0 & \text{otherwise} \end{cases}$$

- so we can calculate the expected value of that feature:

$$\tilde{p}(f) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Feature Function

For each constraint we know, we create a so called feature function:

- For instance, if in the training data *April* is the word following *in*, the translation of *in* is *en* in 90%
- we express the feature in a indicator function:

$$f(x,y) = \begin{cases} 1 & \text{if } x = \text{en and April follows in} \\ 0 & \text{otherwise} \end{cases}$$

- so we can calculate the expected value of that feature:

$$\tilde{p}(f) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Constraint Equation

- also our model $p(y \mid x)$ should correspond to that feature function:

$$p(f) = \sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y)$$

$\tilde{p}(x)$ ... empirical distribution of x in the training data

- expected value p(f) should be the same as in the training data:

$$p(f) = \tilde{p}(f)$$

- which leads to the constraint equation:

$$\sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Constraint Equation

- also our model $p(y \mid x)$ should correspond to that feature function:

$$p(f) = \sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y)$$

$\tilde{p}(x)$ ... empirical distribution of x in the training data

- expected value p(f) should be the same as in the training data:

$$p(f) = \tilde{p}(f)$$

- which leads to the constraint equation:

$$\sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Constraint Equation

- also our model $p(y \mid x)$ should correspond to that feature function:

$$p(f) = \sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y)$$

$\tilde{p}(x)$ ... empirical distribution of x in the training data

- expected value p(f) should be the same as in the training data:

$$p(f) = \tilde{p}(f)$$

- which leads to the constraint equation:

$$\sum_{x,y} \tilde{p}(x) p(y \mid x) f(x,y) = \sum_{x,y} \tilde{p}(x,y) f(x,y)$$

Possible Models

- with the given feature functions f_i, we can define a subset C out of all possible probability functions P, where our model p should be:
- $$C \equiv \{p \in P \mid p(f_i) = \tilde{p}(f_i) \text{ for i} \in \{1, 2, ..., n\}\}$$
- among the models $p \in C$ the ME-philosophy dictates that we select the most uniform distribution - but what does "uniform" mean ?

Possible Models

- with the given feature functions f_i, we can define a subset C out of all possible probability functions P, where our model p should be:
- $$C \equiv \{p \in P \mid p(f_i) = \tilde{p}(f_i) \text{ for i} \in \{1, 2, ..., n\}\}$$
- among the models $p \in C$ the ME-philosophy dictates that we select the most uniform distribution - but what does "uniform" mean ?

Possible Models

- with the given feature functions f_i, we can define a subset C out of all possible probability functions P, where our model p should be:
- $$C \equiv \{p \in P \mid p(f_i) = \tilde{p}(f_i) \text{ for i} \in \{1, 2, ..., n\}\}$$
- among the models $p \in C$ the ME-philosophy dictates that we select the most uniform distribution - but what does “uniform” mean ?

Information Entropy

- consider a discrete probability distribution among m exclusive propositions
  - most informative distribution would occur, when one propositions is true - information entropy would be zero
  - least informative distributionis, when there is no reason to favor any - the only reasonable probability distribution would be uniform - thus the entropy would be maximum (log m)
- the conditional entropy is defined as:

$$H(p) = -\sum_{x,y} \tilde{p}(x,y)p(y \mid x)logp(y \mid x)$$

Information Entropy

- consider a discrete probability distribution among m exclusive propositions
  - most informative distribution would occur, when one propositions is true - information entropy would be zero
  - least informative distributionis, when there is no reason to favor any - the only reasonable probability distribution would be uniform - thus the entropy would be maximum (log m)
- the conditional entropy is defined as:

$$H(p) = -\sum_{x,y} \tilde{p}(x,y)p(y \mid x)logp(y \mid x)$$

Information Entropy

- consider a discrete probability distribution among m exclusive propositions
  - most informative distribution would occur, when one propositions is true - information entropy would be zero
  - least informative distributionis, when there is no reason to favor any - the only reasonable probability distribution would be uniform - thus the entropy would be maximum (log m)
- the conditional entropy is defined as:

$$H(p) = -\sum_{x,y} \tilde{p}(x,y)p(y \mid x)logp(y \mid x)$$

Information Entropy

- consider a discrete probability distribution among m exclusive propositions
  - most informative distribution would occur, when one propositions is true - information entropy would be zero
  - least informative distributionis, when there is no reason to favor any - the only reasonable probability distribution would be uniform - thus the entropy would be maximum (log m)
- the conditional entropy is defined as:

$$H(p) = -\sum_{x,y} \tilde{p}(x,y) p(y \mid x) log p(y \mid x)$$

Maximum Entropy Principle

- the information entropy can therefore be seen as a numerical measure which describes how uninformative a particular probability distribution is
- so to select our model, we choose the model $p_* \in C$ with maximum entropy $H(p)$:

$$p_* = \arg\max_{p \in C} H(p)$$

Maximum Entropy Principle

- the information entropy can therefore be seen as a numerical measure which describes how uninformative a particular probability distribution is
- so to select our model, we choose the model $p_* \in C$ with maximum entropy $H(p)$:

$$p_* = \arg\max_{p \in C} H(p)$$

Lagrange Multipliers

Solving the ME-principle introduces a problem of constrained optimization and therefore uses the method of Lagrange multipliers:

- find

$$p_* = \arg\max_{p \in C} \left\{ -\sum_{x,y} \tilde{p}(x,y) p(y \mid x) log p(y \mid x) \right\}$$

- for each feature f_i (= a constraint) we introduce a parameter λ_i (the Lagrange multiplier)
- so we can calculate a maximum:

$$p_\lambda(y \mid x) = \frac{1}{Z_\lambda(x)} exp\left(\sum_i \lambda_i f_i\right)$$

Lagrange Multipliers

Solving the ME-principle introduces a problem of constrained optimization and therefore uses the method of Lagrange multipliers:

- find

$$p_* = \arg\max_{p \in C} \left\{ -\sum_{x,y} \tilde{p}(x,y) p(y \mid x) log p(y \mid x) \right\}$$

- for each feature f_i (= a constraint) we introduce a parameter λ_i (the Lagrange multiplier)
- so we can calculate a maximum:

$$p_\lambda(y \mid x) = \frac{1}{Z_\lambda(x)} exp(\sum_i \lambda_i f_i)$$

Lagrange Multipliers

Solving the ME-principle introduces a problem of constrained optimization and therefore uses the method of Lagrange multipliers:

- find

$$p_* = \arg\max_{p \in C} \left\{ -\sum_{x,y} \tilde{p}(x,y) p(y \mid x) log p(y \mid x) \right\}$$

- for each feature f_i (= a constraint) we introduce a parameter λ_i (the Lagrange multiplier)
- so we can calculate a maximum:

$$p_\lambda(y \mid x) = \frac{1}{Z_\lambda(x)} exp(\sum_i \lambda_i f_i)$$

Lagrange Multipliers

- where $Z_\lambda(x)$ is a normalizing constant, which can be calculated with the constraint that:

$$\sum_y p_\lambda(y \mid x) = 1$$

- so we come to the so called *Zustandssumme*:

$$Z_\lambda(x) = \sum_y exp(\sum_i \lambda_i f_i)$$

Lagrange Multipliers

- where $Z_\lambda(x)$ is a normalizing constant, which can be calculated with the constraint that:

$$\sum_y p_\lambda(y \mid x) = 1$$

- so we come to the so called *Zustandssumme*:

$$Z_\lambda(x) = \sum_y exp(\sum_i \lambda_i f_i)$$

Calculating the Lagrange Multipliers

- the values of the Lagrange multipliers λ_i can be calculated with the following constraint:

$$p(f) = -\frac{\partial}{\partial \lambda_k} \log Z_\lambda(x)$$

- these m simultaneous equations do not generally possess a closed form solution, and are usually solved by numerical methods - e.g. with the *iterative scaling algorithm*

Calculating the Lagrange Multipliers

- the values of the Lagrange multipliers λ_i can be calculated with the following constraint:

$$p(f) = -\frac{\partial}{\partial \lambda_k} \log Z_\lambda(x)$$

- these m simultaneous equations do not generally possess a closed form solution, and are usually solved by numerical methods - e.g. with the *iterative scaling algorithm*

Maximum Likelihood

- Maximum Likelihood-Ratio requires assumptions about the distribution of a model:
  - you assume a null-hypothesis H_0
  - create a likelihood-ratio to test H_0
- the likelihood approach is most useful when one has lot's of data, but no other prior information (= constraints) about the process

Maximum Likelihood

- Maximum Likelihood-Ratio requires assumptions about the distribution of a model:
  - you assume a null-hypothesis H_0
  - create a likelihood-ratio to test H_0
- the likelihood approach is most useful when one has lot's of data, but no other prior information (= constraints) about the process

Maximum Likelihood

- Maximum Likelihood-Ratio requires assumptions about the distribution of a model:
  - you assume a null-hypothesis H_0
  - create a likelihood-ratio to test H_0
- the likelihood approach is most useful when one has lot's of data, but no other prior information (= constraints) about the process

Maximum Likelihood

- Maximum Likelihood-Ratio requires assumptions about the distribution of a model:
  - you assume a null-hypothesis H_0
  - create a likelihood-ratio to test H_0
- the likelihood approach is most useful when one has lot's of data, but no other prior information ($=$ constraints) about the process

Maximum Likelihood vs. Maximum Entropy

- Maximum-Entropy inference encodes prior information as constraints on the set of possible models and estimates the parameters that make the fewest additional assumptions
- the ME approach is most useful when one has relevant prior information but no appreciable noise in the data
- Maximum Likelihood and Maximum Entropy represent opposite extremes of reasoning, each appropriate to a distinct class of problems.

Maximum Likelihood vs. Maximum Entropy

- Maximum-Entropy inference encodes prior information as constraints on the set of possible models and estimates the parameters that make the fewest additional assumptions
- the ME approach is most useful when one has relevant prior information but no appreciable noise in the data
- Maximum Likelihood and Maximum Entropy represent opposite extremes of reasoning, each appropriate to a distinct class of problems.

Maximum Likelihood vs. Maximum Entropy

- Maximum-Entropy inference encodes prior information as constraints on the set of possible models and estimates the parameters that make the fewest additional assumptions
- the ME approach is most useful when one has relevant prior information but no appreciable noise in the data
- Maximum Likelihood and Maximum Entropy represent opposite extremes of reasoning, each appropriate to a distinct class of problems.

Motivation

In this section a method for automatically selecting features to be included in a ME-model is propsed:

- we begin by specifying a very large collection F of candidate features
- only a subset S of F will be included in our model - the active features
- S should capture as much information about the random process as possible

Motivation

In this section a method for automatically selecting features to be included in a ME-model is propsed:

- we begin by specifying a very large collection F of candidate features
- only a subset S of F will be included in our model - the active features
- S should capture as much information about the random process as possible

Motivation

In this section a method for automatically selecting features to be included in a ME-model is propsed:

- we begin by specifying a very large collection F of candidate features
- only a subset S of F will be included in our model - the active features
- S should capture as much information about the random process as possible

Motivation

In this section a method for automatically selecting features to be included in a ME-model is propsed:

- we begin by specifying a very large collection F of candidate features
- only a subset S of F will be included in our model - the active features
- S should capture as much information about the random process as possible

How to find the Active Features

An incremental approach is used to find the features S:

- each step, one additional features f is added to S and thus is an additional constraint - so the number of possible models decrease
- the choice of which feature to add is determined by the training data
- "adding" a feature means, that the set of allowable models all satisfy the equation $\tilde{p}(f) = p(f)$

How to find the Active Features

An incremental approach is used to find the features S:

- each step, one additional features f is added to S and thus is an additional constraint - so the number of possible models decrease
- the choice of which feature to add is determined by the training data
- "adding" a feature means, that the set of allowable models all satisfy the equation $\tilde{p}(f) = p(f)$

How to find the Active Features

An incremental approach is used to find the features S:

- each step, one additional features f is added to S and thus is an additional constraint - so the number of possible models decrease
- the choice of which feature to add is determined by the training data
- "adding" a feature means, that the set of allowable models all satisfy the equation $\tilde{p}(f) = p(f)$

How to find the Active Features

An incremental approach is used to find the features S:

- each step, one additional features f is added to S and thus is an additional constraint - so the number of possible models decrease
- the choice of which feature to add is determined by the training data
- "adding" a feature means, that the set of allowable models all satisfy the equation $\tilde{p}(f) = p(f)$

Formal Description

- every stage of the incremental algorithm is characterized by a set of active features S, these determine a space of models $C(S)$:

$$C(S) \equiv \{p \in P \mid p(f) = \tilde{p}(f) \text{ for all } f \in S\}$$

- optimal model in this space:

$$p_S \equiv \arg\max_{p \in C(S)} H(p)$$

Formal Description

- every stage of the incremental algorithm is characterized by a set of active features S, these determine a space of models $C(S)$:

$$C(S) \equiv \{p \in P \mid p(f) = \tilde{p}(f) \text{ for all } f \in S\}$$

- optimal model in this space:

$$p_S \equiv \arg \max_{p \in C(S)} H(p)$$

Formal Description

- adding a new feature $\hat{f}$ to S, we get a new set of active features $S \cup \hat{f}$ - so determines a new set of models :

$$C(S \cup \hat{f}) \equiv \left\{ p \in P \mid p(f) = \tilde{p}(f) \text{ for all } f \in S \cup \hat{f} \right\}$$

- now the optimal model is:

$$p_{S \cup \hat{f}} \equiv \arg \max_{p \in C(S \cup \hat{f})} H(p)$$

Formal Description

- adding a new feature $\hat{f}$ to S, we get a new set of active features $S \cup \hat{f}$ - so determines a new set of models :

$$C(S \cup \hat{f}) \equiv \left\{ p \in P \mid p(f) = \tilde{p}(f) \text{ for all } f \in S \cup \hat{f} \right\}$$

- now the optimal model is:

$$p_{S \cup \hat{f}} \equiv \arg \max_{p \in C(S \cup \hat{f})} H(p)$$

Which new Feature

- to decide which new feature $\hat{f}$ we should add, we calculate the log-likelihood ratio with the training data:

$$\Delta L(S, \hat{f}) \equiv L(p_{S \cup \hat{f}}) - L(p_S)$$

- at each step our goal is to select the feature $\hat{f}$ which maximizes the gain $\Delta L(S, \hat{f})$ - thus produces the greatest increase in likelihood of the training sample

Which new Feature

- to decide which new feature $\hat{f}$ we should add, we calculate the log-likelihood ratio with the training data:

$$\Delta L(S, \hat{f}) \equiv L(p_{S \cup \hat{f}}) - L(p_S)$$

- at each step our goal is to select the feature $\hat{f}$ which maximizes the gain $\Delta L(S, \hat{f})$ - thus produces the greatest increase in likelihood of the training sample

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
    - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
    - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

The Algorithm

Input: collection F of candidate features; empirical distribution $\tilde{p}(x, y)$
Output: set S of active features; model p_S incorporating these features

1. start with $S = 0$, thus p_S is uniform
2. do for each candidate feature $f \in F$
   - compute model $p_{S \cup \hat{f}}$ as descriped in section 2
   - compute the gain in the log-likelihood
3. select feature $\hat{f}$ with maximal gain $\Delta L(S, \hat{f})$
4. check termination condition: if $\hat{f}$ leads to an increase in likelihood
5. adjoin $\hat{f}$ to S and compute p_S
6. go to step 2

Performance Problem

- problem of the algorithm:
  for each candidate feature f we must compute the according ME-model, which is computationally costly (see step 2)
- solution:
  we calculate an approximation of $\Delta L(S, f)$, which will be called $\backsim \Delta L(S, f)$

Performance Problem

- problem of the algorithm:
  for each candidate feature f we must compute the according ME-model, which is computationally costly (see step 2)
- solution:
  we calculate an approximation of $\Delta L(S, f)$, which will be called $\backsim \Delta L(S, f)$

Approximate Gains

- model p_S has a set of parameters λ
  model $p_{S \cup f}$ has an additional new parameter α, corresponding to f
- problem: when new parameter α is added, all parameters λ must be recalculated
- solution: make the approximation, that the addition of a feature f affects only α - so it results in a simple one-dimensional optimization problem

Approximate Gains

- model p_S has a set of parameters λ
  model $p_{S \cup f}$ has an additional new parameter α, corresponding to f
- problem: when new parameter α is added, all parameters λ must be recalculated
- solution: make the approximation, that the addition of a feature f affects only α - so it results in a simple one-dimensional optimization problem

Approximate Gains

- model p_S has a set of parameters λ
  model $p_{S \cup f}$ has an additional new parameter α, corresponding to f
- problem: when new parameter α is added, all parameters λ must be recalculated
- solution: make the approximation, that the addition of a feature f affects only α - so it results in a simple one-dimensional optimization problem

Approximate Gains

- best model containing features $S \cup f$ has the form:

$$p^{\alpha}_{S,f} = \frac{1}{Z_{\alpha}(x)} p_s(y \mid x) e^{\alpha f(x,y)}$$

- the approximate gain of parameter f:

$$\begin{aligned} G_{S,f}(\alpha) &= L(p^{\alpha}_{S,f}) - L(p_S) \\ &= -\sum_x \tilde{p}(x) log Z_{\alpha}(x) + \alpha \tilde{p}(f) \end{aligned}$$

Approximate Gains

- best model containing features $S \cup f$ has the form:

$$p^{\alpha}_{S,f} = \frac{1}{Z_{\alpha}(x)} p_s(y \mid x) e^{\alpha f(x,y)}$$

- the approximate gain of parameter f:

$$\begin{aligned} G_{S,f}(\alpha) &= L(p^{\alpha}_{S,f}) - L(p_S) \\ &= -\sum_x \tilde{p}(x) log Z_{\alpha}(x) + \alpha \tilde{p}(f) \end{aligned}$$

Approximated Likelihood

- so the approximated likelihood-ratio is:

$$\backsim \Delta L(S, f) = \max_{\alpha} G_{S,f}(\alpha)$$

- and the optimal model:

$$\backsim p_{S \cup f} = \arg\max_{p^{\alpha}_{S,f}} G_{S,f}(\alpha)$$

- this one-dimensional optimization problem can be solved by any popular line-search technique, such as Newton's method

Approximated Likelihood

- so the approximated likelihood-ratio is:

$$\backsim \Delta L(S, f) = \max_{\alpha} G_{S,f}(\alpha)$$

- and the optimal model:

$$\backsim p_{S \cup f} = \arg\max_{p^{\alpha}_{S,f}} G_{S,f}(\alpha)$$

- this one-dimensional optimization problem can be solved by any popular line-search technique, such as Newton's method

Approximated Likelihood

- so the approximated likelihood-ratio is:

$$\backsim \Delta L(S,f) = \max_{\alpha} G_{S,f}(\alpha)$$

- and the optimal model:

$$\backsim p_{S\cup f} = \arg\max_{p^{\alpha}_{S,f}} G_{S,f}(\alpha)$$

- this one-dimensional optimization problem can be solved by any popular line-search technique, such as Newton's method

French-to-English Example

- a automatic French-to-English machine translation system is given as an example
- several applications of maximum entropy modeling will be discussed, within Candide - a system developed at IBM

French-to-English Example

- a automatic French-to-English machine translation system is given as an example
- several applications of maximum entropy modeling will be discussed, within Candide - a system developed at IBM

Statistical Translation

First a short review of "traditional" statistical translation:

- translation from an French sentence F to an most likely English sentence $\hat{E}$:

$$\begin{aligned} \hat{E} &= \arg\max_E p(E \mid F) \\ &= \arg\max_E p(F \mid E) p(E) \text{ (Bayes' theorem)} \end{aligned}$$

- $p(E)$ ... language model
  $p(F \mid E)$ ... translation model

Statistical Translation

First a short review of "traditional" statistical translation:

- translation from an French sentence F to an most likely English sentence $\hat{E}$:

$$\begin{aligned} \hat{E} &= \arg\max_E p(E \mid F) \\ &= \arg\max_E p(F \mid E) p(E) \text{ (Bayes' theorem)} \end{aligned}$$

- $p(E)$ ... language model
  $p(F \mid E)$ ... translation model

Translation Model

- for the translation model $p(F \mid E)$ we get an alignment A between the French and English words:

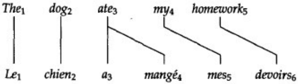


Figure 4
Alignment of a French–English sentence pair. The subscripts give the position of each word in its sentence. Here $a_1 = 1$, $a_2 = 2$, $a_3 = a_4 = 3$, $a_5 = 4$, and $a_6 = 5$.

- so $p(F \mid E)$ can be expressed as the sum over all possible alignments A between E and F, of the probability of F and A given E:

$$p(F \mid E) = \sum_{A} p(F, A \mid E)$$

Translation Model

- for the translation model $p(F \mid E)$ we get an alignment A between the French and English words:

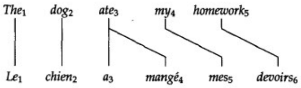


Figure 4
Alignment of a French–English sentence pair. The subscripts give the position of each word in its sentence. Here $a_1 = 1$, $a_2 = 2$, $a_3 = a_4 = 3$, $a_5 = 4$, and $a_6 = 5$.

- so $p(F \mid E)$ can be expressed as the sum over all possible alignments A between E and F, of the probability of F and A given E:

$$p(F \mid E) = \sum_A p(F, A \mid E)$$

Viterbi Alignment

- for computational reasons we make the assumption, that there exists only one extremely probable alignment $\hat{A}$, the *Viterbi Alignment*, for which:

$$p(F \mid E) \approx p(F, \hat{A} \mid E)$$

- the basic translation model is given by:

$$p(F, A \mid E) = \prod_{i=1}^{|E|} p(n(e_i) \mid e_i) \prod_{j=1}^{|F|} p(y_j \mid e_{a_j}) d(A \mid E, F)$$

$p(n(e_i) \mid e_i)$ ... e generates n French words
$d(A \mid E, F)$ ... order of French words

Viterbi Alignment

- for computational reasons we make the assumption, that there exists only one extremely probable alignment $\hat{A}$, the *Viterbi Alignment*, for which:

$$p(F \mid E) \approx p(F, \hat{A} \mid E)$$

- the basic translation model is given by:

$$p(F, A \mid E) = \prod_{i=1}^{|E|} p(n(e_i) \mid e_i) \prod_{j=1}^{|F|} p(y_j \mid e_{aj}) d(A \mid E, F)$$

$p(n(e_i) \mid e_i)$ ... e generates n French words
$d(A \mid E, F)$ ... order of French words

Training

- An EM-algorithm can be used to estimate the parameters of this basic translation model, so that it maximizes some bilingual corpus (here from the Canadian Parliament).
- probabilities for the translation of the English word *in*:

Translation	Probability
dans	0.3004
à	0.2275
de	0.1428
en	0.1361
pour	0.0349
(OTHER)	0.0290
au cours de	0.0233
	0.0154
sur	0.0123
par	0.0101
pendant	0.0044

Context

- the previous model has one major shortcome: it does not take the English context into account
- therefore a maximum entropy model $p_e(y \mid x)$ for each English word e is used
- $p_e(y \mid x)$ represents the probability that an translator would choose y as the French translation of e, given the surrounding English context x

Context

- the previous model has one major shortcome: it does not take the English context into account
- therefore a maximum entropy model $p_e(y \mid x)$ for each English word e is used
- $p_e(y \mid x)$ represents the probability that an translator would choose y as the French translation of e, given the surrounding English context x

Context

- the previous model has one major shortcome: it does not take the English context into account
- therefore a maximum entropy model $p_e(y \mid x)$ for each English word e is used
- $p_e(y \mid x)$ represents the probability that an translator would choose y as the French translation of e, given the surrounding English context x

Features

- now, in our example of the translation of *in*, x contains the six words surrounding *in*
- so we can define some candidate features:

$$f_1(x,y) = \begin{cases} 1 & \text{if } y = \textit{en} \text{ and } \textit{April} \in \square\square\square\boxed{\bullet}\square\square \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(x,y) = \begin{cases} 1 & \text{if } y = \textit{pendant} \text{ and } \textit{weeks} \in \square\square\square\boxed{\bullet}\boxed{\bullet}\boxed{\bullet} \\ 0 & \text{otherwise} \end{cases}$$

Feature Templates

Because the set of possible features is very big, the authors restricted them to the following five feature templates:

Feature templates for word-translation modeling. $|\mathcal{V}_{\mathcal{E}}|$ is the size of the English vocabulary; $|\mathcal{V}_{\mathcal{F}}|$ the size of the French vocabulary.

Template	Number of Actual Features	$f(x,y) = 1$ if and only if ...
1	$\lvert\mathcal{V}_{\mathcal{F}}\rvert$	$y = \diamond$
2	$\lvert\mathcal{V}_{\mathcal{F}}\rvert \cdot \lvert\mathcal{V}_{\mathcal{E}}\rvert$	$y = \diamond$ and $\square \in$ [•]
3	$\lvert\mathcal{V}_{\mathcal{F}}\rvert \cdot \lvert\mathcal{V}_{\mathcal{E}}\rvert$	$y = \diamond$ and $\square \in$ [•]
4	$\lvert\mathcal{V}_{\mathcal{F}}\rvert \cdot \lvert\mathcal{V}_{\mathcal{E}}\rvert$	$y = \diamond$ and $\square \in$ [• • •]
5	$\lvert\mathcal{V}_{\mathcal{F}}\rvert \cdot \lvert\mathcal{V}_{\mathcal{E}}\rvert$	$y = \diamond$ and $\square \in$ [• • •]

where $\diamond$ is a French and $\square$ is an English word

Automatic Feature Selection

Maximum entropy model to predict French translation of *in*. Features shown here were the first features selected not from template 1. [*verb marker*] denotes a morphological marker inserted to indicate the presence of a verb as the next word.

	Feature $f(x,y)$						$\sim\Delta L(\mathcal{S},f)$	$L(p)$
y=*à* and *Canada* ∈				•			0.0415	−2.9674
y=*à* and *House* ∈				•			0.0361	−2.9281
y=*en* and *the* ∈				•			0.0221	−2.8944
y=*pour* and *order* ∈				•			0.0224	−2.8703
y=*dans* and *speech* ∈				•	•	•	0.0190	−2.8525
y=*dans* and *area* ∈				•	•	•	0.0153	−2.8377
y=*de* and *increase* ∈	•	•	•				0.0151	−2.8209
y=[*verb marker*] and *my* ∈				•			0.0141	−2.8034
y=*dans* and *case* ∈				•	•	•	0.0116	−2.7918
y=*au cours de* and *year* ∈				•	•	•	0.0104	−2.7792

Translation Model

- the ME word translation model has to be incorporated into the translation model $p(F \mid E)$
- this means the context-independendt model $p(y \mid x)$ has to be replaced with $p_e(y \mid x)$:

$$p(F, A \mid E) = \prod_{i=1}^{|E|} p(n(e_i) \mid e_i) \prod_{j=1}^{|F|} p_{e_{aj}}(y_j \mid x_{aj}) d(A \mid E, F)$$

where x_{aj} is the context of the English word e_{aj}

Translation Model

- the ME word translation model has to be incorporated into the translation model $p(F \mid E)$
- this means the context-independedt model $p(y \mid x)$ has to be replaced with $p_e(y \mid x)$:

$$p(F, A \mid E) = \prod_{i=1}^{|E|} p(n(e_i) \mid e_i) \prod_{j=1}^{|F|} p_{e_{aj}}(y_j \mid x_{aj}) d(A \mid E, F)$$

where x_{aj} is the context of the English word e_{aj}

Segmentation

- since processing time is exponential in the length of the input sentence, the French sentences have to be splitted into smaller parts
- task is to find a safe position at which to split
- in our case, a safe segmentation is dependent on the Viterbi alignment $\hat{A}$

Segmentation

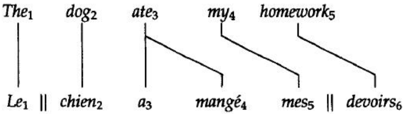
- since processing time is exponential in the length of the input sentence, the French sentences have to be splitted into smaller parts
- task is to find a safe position at which to split
- in our case, a safe segmentation is dependent on the Viterbi alignment $\hat{A}$

Segmentation

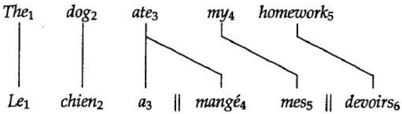
- since processing time is exponential in the length of the input sentence, the French sentences have to be splitted into smaller parts
- task is to find a safe position at which to split
- in our case, a safe segmentation is dependent on the Viterbi alignment $\hat{A}$

Save Segmentation

- a position of a safe segmentation is called a *rift*, e.g.:



- whereas the following would be a unsafe segmentation:



because a word in the translated sentence is aligned to words in two different segments of the input sentence

Segmentation Algorithm

- now a ME-model assigns to each location in the French sentence a score $p(\mathit{rift} \mid x)$
- then a dynamic programming algorithm selects the optimal splitting of the sentence, so that no segment contains more than 10 words
- these segments are not logically coherent, but can be translated sequentially from left to right

Segmentation Algorithm

- now a ME-model assigns to each location in the French sentence a score $p(rift \mid x)$
- then a dynamic programming algorithm selects the optimal splitting of the sentence, so that no segment contains more than 10 words
- these segments are not logically coherent, but can be translated sequentially from left to right

Segmentation Algorithm

- now a ME-model assigns to each location in the French sentence a score $p(rift \mid x)$
- then a dynamic programming algorithm selects the optimal splitting of the sentence, so that no segment contains more than 10 words
- these segments are not logically coherent, but can be translated sequentially from left to right

System's Segmentation

An example of the system's segmentation:

Monsieur l'Orateur

,

j'aimerais poser une question au
Ministre des Transports.

—

A quelle date le
nouveau règlement devrait il entrer en vigeur?

—

Quels furent les critères utilisés
pour l'évaluation
de ces biens.

—

Nous
savons que si nous pouvions contrôler la folle avoine
dans l'ouest du Canada, en
un an nous
augmenterions notre rendement en
céréales de 1 milliard de dollars.

Word Reordering

- the English word order is often very different from the French one
- input French sentences are shuffled in a preprocessing stage into a order more closely to the English word order

Word Reordering

- the English word order is often very different from the French one
- input French sentences are shuffled in a preprocessing stage into a order more closely to the English word order

NOUN de NOUN

French phrases which have the *NOUN de NOUN* form are sometimes changed in the English translation:

NOUN *de* NOUN phrases and their English equivalents.

Word-for-word Phrases	
somme d'argent	*sum of money*
pays d'origin	*country of origin*
question de privilège	*question of privilege*
conflit d'intérêt	*conflict of interest*
Interchanged Phrases	
bureau de poste	*post office*
taux d'intérêt	*interest rate*
compagnie d'assurance	*insurance company*
gardien de prison	*prison guard*

Maximum Entropy Model

- ME-model decides, given a French *NOUN de NOUN* phrase, if the nouns should be interchanged in the English translation
- *y=no-interchange*, if the English translation is a word-for-word translation, otherwise *y=interchange*
- candidate features are taken from a template (next slide)

Maximum Entropy Model

- ME-model decides, given a French *NOUN de NOUN* phrase, if the nouns should be interchanged in the English translation
- *y=no-interchange*, if the English translation is a word-for-word translation, otherwise *y=interchange*
- candidate features are taken from a template (next slide)

Maximum Entropy Model

- ME-model decides, given a French *NOUN de NOUN* phrase, if the nouns should be interchanged in the English translation
- *y=no-interchange*, if the English translation is a word-for-word translation, otherwise *y=interchange*
- candidate features are taken from a template (next slide)

Feature Template

- $\square_1$ and $\square_2$ are the French words, $\diamondsuit$ means *interchange* or *no-interchange*:

Template features for NOUN *de* NOUN model.

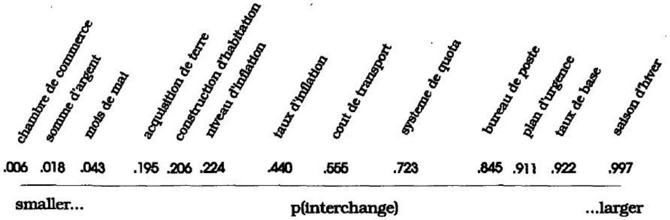
Template	Number of Actual Features	$f(x,y) = 1$ if and only if ...		
1	$2\|\mathcal{V}_{\mathcal{F}}\|$	$y = \diamondsuit$	and	$\text{NOUN}_L = \square$
2	$2\|\mathcal{V}_{\mathcal{F}}\|$	$y = \diamondsuit$	and	$\text{NOUN}_R = \square$
3	$2\|\mathcal{V}_{\mathcal{F}}\|^2$	$y = \diamondsuit$	and	$\text{NOUN}_L = \square_1$ and $\text{NOUN}_R = \square_2$

- e.g. temlate 1 features consider only the left noun:

$$f(x,y) = \begin{cases} 1 & \text{if y=interchange and left NOUN=système} \\ 0 & \text{otherwise} \end{cases}$$

Maximum Entropy Model

- feture-selection algorithm is used to construct a ME-model
- here some examples, if *p(interchange)* > 0.5, the nouns are interchanged:



References

- Adam L. Berger, Stephen A. Della Pietra, Vincent J. Della Pietra; *A Maximum Entropy Approach to Natural Language Processing*; 1996 Association for Computational Linguistics
- Edward Schofield; *Fitting maximum-entropy models on large sample spaces*; 2006, Dissertation Imperial College London
- Wikipedia; *Principle of maximum entropy*; http://en.wikipedia.org/wiki/Maximum_entropy; (accessed 6. December 2006)
- Wikipedia; *Lagrange multipliers*; http://en.wikipedia.org/wiki/Lagrange_multipliers; (accessed 6. December 2006)
- Wikipedia; *Likelihood-Quotienten-Test*; http://de.wikipedia.org/wiki/Likelihood-Quotienten-Test; (accessed 6. December 2006)