CA-4012 Machine Translation

Introduction To Statistical Model for Machine Translation

Dr. Kolawole Adebayo
ADAPT Centre School of Computing, Dublin City University

Engaging Content
Engaging People

MT Evaluation- A Recap

- Adequacy, fluency, comprehensibility
- Human evaluation vs Automatic Evaluation
- Automatic Evaluation – Similarity (matching) Vs Difference(edit distance)
- Unigram vs N-grams
- Metrics – Precision, recall, F-measure
- BLEU, METEOR, Word Error Rate (WER)

Engaging Content
Engaging People

Overview

- Statistical machine translation (SMT)
- Translation model
- Language Model

Statistical Machine Translation (SMT)

An automated process that statistically generates the best possible target language sentence ***t*** for the given source language sentence ***s***.

The best possible target language sentence is defined as the target sentence with the highest probability given the source language sentence.

both SMT and NMT are probabilistic approaches*

Statistical Machine Translation (SMT)

This conditional probability is defined as

P(t/s)

where,

t ☐ target-language sentence, and

s ☐ source-language sentence

P(t/s) can be interpreted as the probability of generating the target language sentence t given the source sentence s

In order to find the best target language sentence tˆ among many others, we have to look for the maximal probability P(t|s):

$$\hat{t} = \arg\max_{t} P(t|s)$$

Generative Model for SMT

Using Bayes rule:

$$P(t|s) = \frac{P(t) \cdot P(s|t)}{P(s)}$$

Search for the maximal probability P(t|s) becomes

$$\hat{t} = \arg\max_t \frac{P(t) \cdot P(s|t)}{P(s)}$$

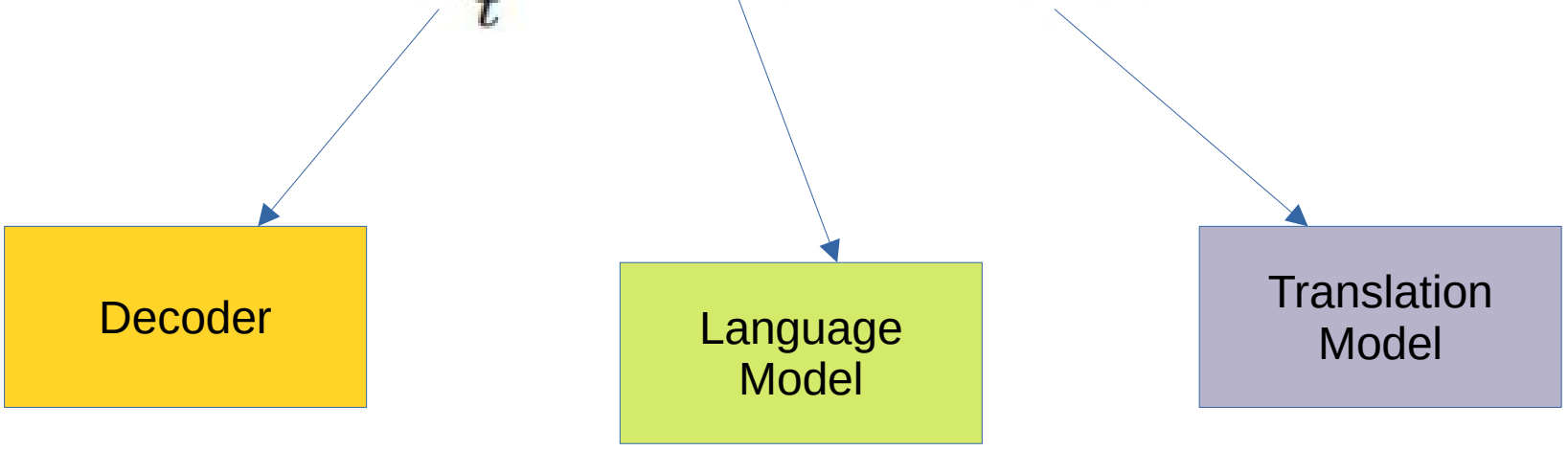
where, P(s) □ the probability of the source sentence

P(s) is fixed and so has no influence on the maximal probability

Generative Model for SMT

New equation:

$$\hat{t} = \arg\max_{t} P(t) \cdot P(s|t)$$



Generative Model for SMT

P(t) □ language model

□ models probability of the target sentence ***t*** controlling the structure of the generated target sentence

□ learns from monolingual texts in the target language.

P(s|t) □ translation model

□models probability that the source sentence ***s*** is a translation of the target sentence ***t***.

□learns from parallel texts

Generative Model for SMT

- Generative models focus on underlying distribution of given data points.
- Models can then determine better target sentence or **generate** new data instance.
- Generative models have good theoretical framework but are not the best option in practice
- Discriminative models are now more mainstream

Discriminative Model for SMT

- tend to learn how to distinguish different classes in a given dataset
- capable of distinguishing classes for existing data (to “discriminate”)

To model the translation probability P(tls) model, we introduce weights λ for each of the two models – λTM for translation model P(slt) and λLM for language model P(t)

Discriminative Model for SMT

$P(t|s) = P(t)^{\lambda_{LM}} P(s|t)^{\lambda_{TM}}$

The weights give more or less importance to each of the models.

In contrast, generative model gives equal importance to each model

Applying rules of exponential function and logarithms

$$P(t|s) = e^{(\lambda_{LM} \cdot log(P(t) + \lambda_{TM} \cdot log(P(s|t)))}$$

Generalised as → $$P(t|s) = \frac{e^{\sum \lambda_i \cdot h_i(s,t)}}{\sum_t e^{\sum \lambda_i \cdot h_i(s,t)}}$$

Discriminative Model for SMT

$$P(t|s) = \frac{e^{\sum \lambda_i \cdot h_i(s,t)}}{\sum_t e^{\sum \lambda_i \cdot h_i(s,t)}}$$

the denominator is constant because it always takes all target sentences into account

⇒ it can be neglected

$$\hat{t} = \arg\max_t \; e^{\sum \lambda_i \cdot h_i(s,t)}$$

discriminative model for MT (also called “log-linear model”)

Discriminative Model for SMT

- tend to learn how to distinguish different classes in a given dataset
- capable of distinguishing classes for existing data (to “discriminate”)
- Formulates MT as a supervised learning problem and reprsents translation with a set of features

To model the translation probability P(tls) model, we introduce weights λ for each of the two models – λTM for translation model P(slt) and λLM for language model P(t)

Translation model: sentence level

Let us consider the following parallel corpus:

French	English
J'aime le garçon .	I love the boy .
J'aime le chien .	I love the dog .
Ils aiment le chien .	They love the dog .
Ils parlent à la fille .	They talk to the girl .
Ils parlent au chien .	They talk to the dog .
Je parle à la mère .	I talk to the mother .

and our new sentence pair:

Je parle au chien . I talk to the dog .

- all the words can be found in the parallel text
- but the entire sentence cannot!

From sentence level to word level

- What can we learn here?
  - there is a huge number of different sentences in all languages
  - Can we rely on sentence probability?
    - Not a good idea
  - Translation model probabilities are estimated on the word level

Language model

The language model (LM) in an SMT systems models the probability of the target language sentence *P(t)*

An LM models probability of a sentence in the given language, where the sentence consists of N words w_n and it is usually denoted as w_1^N

Language model: sentence level

- the probability of a particular number on a dice is 1/6 because there are six possible numbers on the dice

- the probability of a particular target language sentence t is 1/N because there are N possible sentences in this language

From sentence level to word level

- there is a huge number of different sentences in all languages
  - Can we rely on sentence level probability?
    - Not a good idea
  - Language model probabilities are equally estimated on the word level

Language model: word level

the most natural idea:

$$P(SENT) = \prod_{i=1}^{N} p(w_i)$$

where $w_1, w_2, \ldots, w_i, \ldots, w_N$ are words in the sentence SENT

“unigram” language model

Unigram language model

the most natural idea:
Our example (only target language needed):

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

What is the probability of the first sentence “I love the boy .”?

Unigram language model

$$P(word) = \frac{number\ of\ occurences\ of\ the\ word}{total\ number\ of\ words}$$

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

total number of words = **33**
number of occurences of “I” = **3**
number of occurences of “love” = **3**
number of occurences of “the” = **6**
number of occurences of “boy” = **1**
number of occurences of “.” = **6**

Unigram language model

P(“I ”) = 3/33
P(“love”) = 3/33
P(“the”) = 6/33
P(“boy”) = 1/33
P(“.”) = 6/33

P(“I love the boy .”) = 3/33 * 3/33 * 6/33 * 1/33 * 6/33
= 324/39135393 = 0.0002732

Unigram language model

What about the sentence the I boy . love ?

P(“I ”) = 3/33
P(“love”) = 3/33
P(“the”) = 6/33
P(“boy”) = 1/33
P(“.”) = 6/33

$$P(\text{“the I boy . love ”}) = 6/33 * 3/33 * 1/33 * 6/33 * 3/33$$
$$= 324/39135393$$
$$= 0.0002732$$

The curse of Unigram model

$$P(\text{“I love the boy .”}) = 3/33 * 3/33 * 6/33 * 1/33 * 6/33$$
$$= 324/39135393 = 0.0002732$$

$$P(\text{“the I boy . love ”}) = 6/33 * 3/33 * 1/33 * 6/33 * 3/33$$
$$= 324/39135393$$
$$= 0.0002732$$

Same probability!!!

However, we know that:

‘I love the boy.’ != ‘the I boy. Love’ != ‘the boy. I love’

The curse of Unigram model

P(‘I love the boy.’) **should not be** P(‘the I boy. love’)

Why is that so?

- naive assumption that words in a sentence are independent of each other
- dismisses word order and context

Solution?

- incorporate words ‘history’ into probability distribution
- history --> previous words

Unigram model

This probability can be represented as product of probabilities of each word given the history of this word:

$$P(w_1^N) = \prod_{n=1}^{N} P(w_n|h_n)$$

How the history h_n is defined ??

Unigram model

- Taking all preceding words as history would be practically equally difficult as staying at the sentence level

- Neglecting the history and relying on single words ("unigram" language model) is not the best idea

Solution:

- The history h_i usually does not consist of all words in the sentence, but of the $n - 1$ preceding words
- Such model relies on n words (the word itself and its history consisting of n − 1 words)
- It is called the n-gram language model.
- n-gram language models with n usually between 3 and 6 are widely used in SMT systems.

Ngram language model

Calculating n-gram probabilities

$$p(w_i|h_i) = \frac{C(w_i, h_i)}{C(h_i)}$$

$C(w_i, h_i)$ □ the number of occurrences of the word together with its history (n-gram), and

$C(h_i)$ □ the number of occurrences of the history.

To better estimate the probabilities for words at the sentence boundaries, symbols for the beginning <*s*> and the end <*s*> of a sentence are added to each sentence

Bigram language model

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

$$P(\text{“I love the boy .”}) = p(\text{“I ”}) \cdot p(\text{“love” | “I ”}) \cdot p(\text{“the” | “love”}) \cdot p(\text{“boy” | “the”}) \cdot p(\text{“.” | “boy”})$$

Bigram language model

The probability of the first word is:

$$P(\text{“}I\text{”}) = \frac{C(\text{“}I\text{”})}{N_{total}} = \frac{3}{33}$$

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

the first word does not have a history N_{total} is the total number of words in the corpus

Bigram language model

The probability of the following word:

$$P(\text{“love”} \mid \text{“I”}) = \frac{C(\text{“I love”})}{C(\text{“I”})} = \frac{2}{3}$$

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

The probability of the third word:

$$P(\text{“the”} \mid \text{“love”}) = \frac{C(\text{“love the”})}{C(\text{“love”})} = \frac{3}{3} = 1$$

etc.

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

Trigram language model

Example

<s> I love the boy . <\s>
<s> I love the dog . <\s>
<s> They love the dog . <\s>
<s> They talk to the girl . <\s>
<s> They talk to the dog . <\s>
<s> I talk to the mother . <\s>

The trigram (3-gram) language model probability of the first sentence

P(“I love the boy .”)

Trigram language model

P(“I love the boy .”) =

P(“I” | < s >)
· P(“love” | < s > “I”)
· P(“the” | “I love”)
· P(“boy” | “love the”)
· P(“.” | “the boy”)
· P(< \s > | “boy .”)

<s> I love the boy . <\s>	P(“I” | < s >)	=	**3/6**
<s> I love the dog . <\s>	P(“love” | < s > “I”)	=	**2/3**
<s> They love the dog . <\s>	P(“the” | “I love”)	=	**2/2**
<s> They talk to the girl . <\s>	P(“boy” | “love the”)	=	**1/3**
<s> They talk to the dog . <\s>	P(“.” | “the boy”)	=	**1/1**
<s> I talk to the mother . <\s>	P(< \s > | “boy .”)	=	**1/1**

Trigram language model

So that the probability of the sentence will be:

$3/6 \cdot 2/3 \cdot 2/2 \cdot 1/3 \cdot 1/1 \cdot 1/1 =$ **0.11**

Calculate the trigram language model probabilities of the following two sentences which do not appear in the training corpus ("test sentences")

i. *I talk to the girl.*

ii. *I talk to the boy*

Trigram language model

P(“*I talk to the girl.*”) = ?

P(“*I talk to the boy*”) = ?

Unseen n-grams ?

Curse of Ngrams language model

P("boy" | "to the") = 0 ("to the boy" does not exist)
P("*I talk to the boy*") = 0

A common problem that occurs in count-based language models
i.e, if only one n-gram in a sentence does not appear in the training corpus, the probability of the whole sentence is 0.

Partial solution

To train count-based n-gram language models on large amounts of data
Still, there is a number of n-grams which do not appear even in large texts, particularly for larger n

Curse of Ngrams language

Longer n-grams vs shorter n-grams

- longer n-grams cover more context, but their coverage drops (they appear less frequently in texts)
- shorter n-grams cover very limited context, but they are more frequent in texts.

Problem:

No matter how large the training corpus is, if only one n-gram in a test sentence does not appear in the training corpus, the probability of the whole sentence is 0.

Solution:

Smoothing □ assigns non-zero probabilities to unseen n-grams.

Ngrams model - Smoothing

Example:

P(“boy” | “to the”) = 0

because the trigram “*to the boy*” does not exist in the training corpus

To avoid this, the probability is changed to something other than 0

Methods used for smoothing:

I. Add-one
II. Back-off
III. Interpolation

Ngrams model - Smoothing

Add-one:

A very simple solution is to say that each unseen n-gram has actually been seen exactly once, hence "*add one*" smoothing

Example:

C("*to the boy*") is set to to 1 instead of 0.

Now, if we add one trigram to our corpus, the number of possible histories C("the boy") also increases by 1

$$\frac{C(\text{“to the boy”})}{C(\text{“the boy”})+1} = \frac{1}{C(\text{“the boy”})+1}$$

However, the sum of probabilities of all trigrams beginning with "to the" has to be 1

$$\sum_{w} P(w|\text{“to the”}) = 1$$

Where, w are all words seen after "*to the*"

Ngrams model - Smoothing

Example:

In our corpus, the words w seen after “to the” are “girl”, “dog” and “mother”

P(“girl” | “to the)” + P(“dog” | “to the”) + P(“mother” | “to the”) = 1

Now, with one more trigram, their sum will have to be reduced and will become

$$1 - \frac{1}{C(\text{“the boy”}) + 1}$$

Thus, the introduced unseen n-gram is “stealing” a part of the probability from the seen n-grams

Problem:

If there are too many unseen trigrams, they will “steal” more and more probabilities from the seen trigrams

Ngrams model - Smoothing

Back-off

Trusts the highest n-gram order language model which contains the given n-gram

If the n-gram is not seen, then look for n − 1-grams, if still not seen, then look for n − 2-grams, etc., with the unigrams as a last resort

For a trigram language model:

```
if trigram count > 0:
  use it
else:
  if bigram count > 0:
    use it
  else:
    use the unigram count
```

Ngrams model - Smoothing

Interpolation

Combines all n-gram orders

For a trigram LM, the probability becomes

$$P(w_i \mid w_{i-1}, w_{i-2}) = \lambda_1 \cdot P(w_i) + \lambda_2 \cdot P(w_i \mid w_{i-1}) + \lambda_3 \cdot P(w_i \mid w_{i-1}, w_{i-2})$$

where the weights sum to 1

$$\lambda_1 + \lambda_2 + \lambda_3 = 1$$

Ngrams model - Smoothing

Now, the probabilities P("boy"|"to the") and P("girl"|"love the") will not be 0 because bigrams and unigrams are seen in the corpus

One of the most used smoothing methods is the Kneser-Ney method

Kneser-Ney smoothing[1]

- Widely considered the most effective method of smoothing
- Uses absolute discounting by subtracting a fixed value from the probability's lower order terms to omit n-grams with lower frequencies
- Is considered equally effective for both higher and lower order n-grams

[1] https://en.wikipedia.org/wiki/Kneser%E2%80%93Ney_smoothing

Evaluating language model -

It is evaluated by predicting the probability of an unseen sentence $w_1 w_2 \ldots w_N$ by using this language model

The behaviour of a language model is often evaluated directly on its predicted probabilities, by calculating its "perplexity" (PP)

$$PP = 2^{H(LM)}$$

H(LM) □ the cross-entropy of the language model

$$H(LM) = \frac{1}{N} log P_{LM}(w_1^N)$$

PLM(w_1^N) □ the probability of the sentence w_1^N calculated by the language model

Perplexity

- If the perplexity for the given sentence is low, it means that the language model is well capable of predicting its probability
- If it is high, then the model cannot predict well the probability of a sentence (it is “perplexed”)

Example

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

What is the perplexity of test sentence “*I talk to the girl .*”?

Perplexity

- If the perplexity for the given sentence is low, it means that the language model is well capable of predicting its probability
- If it is high, then the model cannot predict well the probability of a sentence (it is “perplexed”)

Example

I love the boy .
I love the dog .
They love the dog .
They talk to the girl .
They talk to the dog .
I talk to the mother .

What is the perplexity of test sentence “*I talk to the girl .*”?

Perplexity

The probability of the sentence will be:

P_{LM} (“I talk to the girl .”) = P(“I” | < s >< \s >)
· P(“talk” | “ < s > I”)
· P(“to” | “I talk”)
· P(“the” | “talk to”)
· P(“girl” | “to the”)
· P(“.” | “the girl”)

The cross-entropy will be:

$$H(LM) = \frac{1}{8} log P_{LM}(\text{“I talk to the girl .”})$$

(N = 8, 8 words including symbols for beginning and end of a sentence)

Finally the perplexity:

$$PP = 2^{H(LM)}$$

Credit

Slides adapted from 2021/2022 lecture by Dr. Pintu Lohar