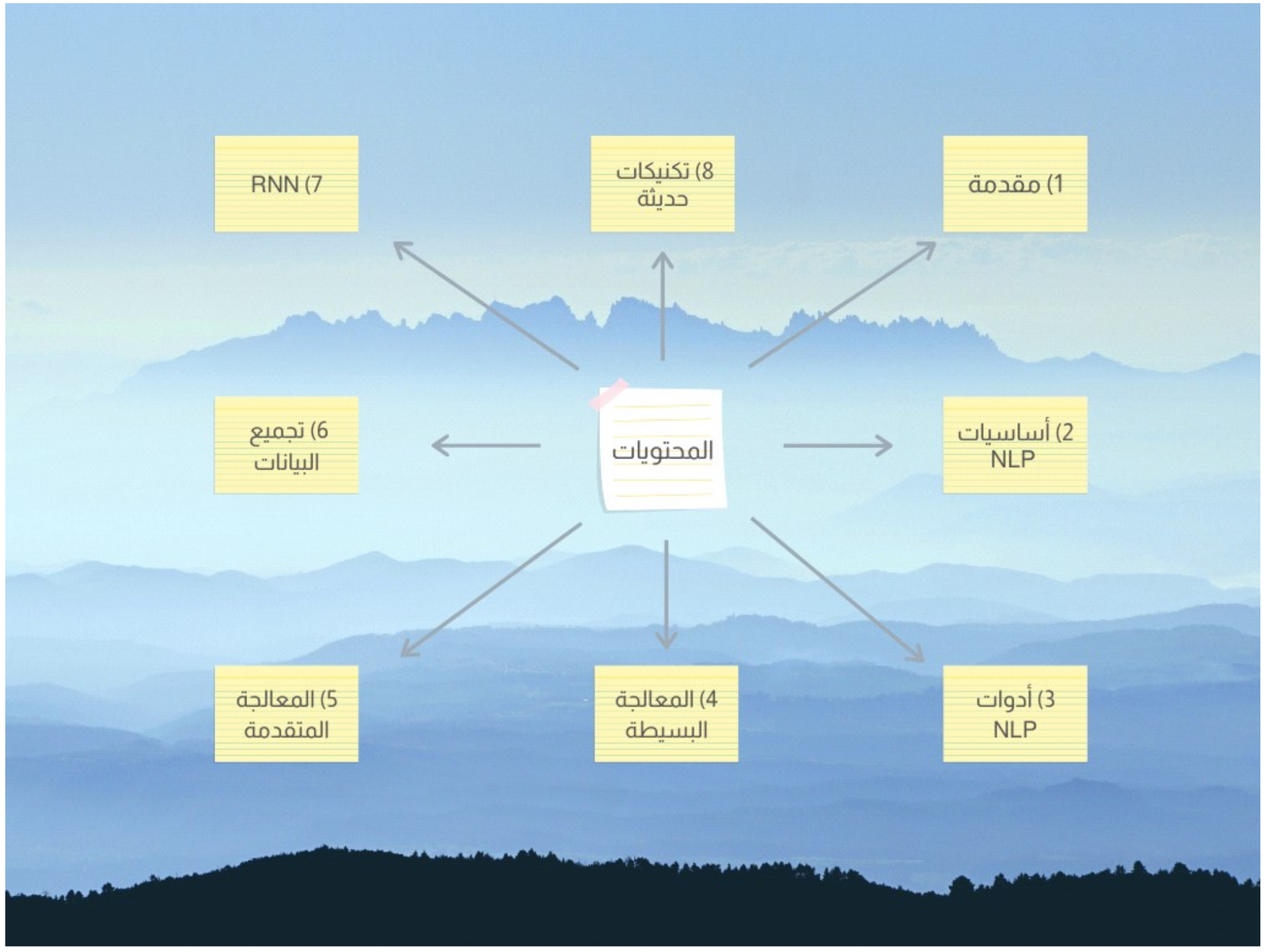
Natural Language Processing المعالجة اللغوية الطبيعية



المحتويات

				التطبيقات	العقبات و التحديات	تاريخ NLP	ما هو NLP	المحتويات	1) مقدمة
					البحث في النصوص	ملفات pdf	الملفات النصية	المكتبات	2) أساسيات NLP
T.Visualization	Syntactic Struc.	Matchers	Stopwords	NER	Stem & Lemm	POS	Sent. Segm.	Tokenization	3) أدوات NLP
	Dist. Similarity	Text Similarity	TF-IDF	BOW	Word2Vec	T. Vectors	Word embed	Word Meaning	4)المعالجة البسيطة
T. Generation	L. Modeling	NGrams	Lexicons	GloVe	NMF	LDA	T. Clustering	T. Classification	5)المعاجلة المتقدمة
	Summarization & Snippets		Ans. Questions		Auto Correct	Vader	Naïve Bayes	Sent. Analysis	
Search Engine	Relative Extraction		Information Retrieval		Information Extraction		Data Scraping	Tweet Collecting	6)تجميع البيانات
					Rec NN\TNN	GRU	LSTM	Seq to Seq	RNN (7
Chat Bot	Gensim	FastText	Bert	Transformer	Attention Model	T. Forcing	CNN	Word Cloud	8)تكنيكات حديثة

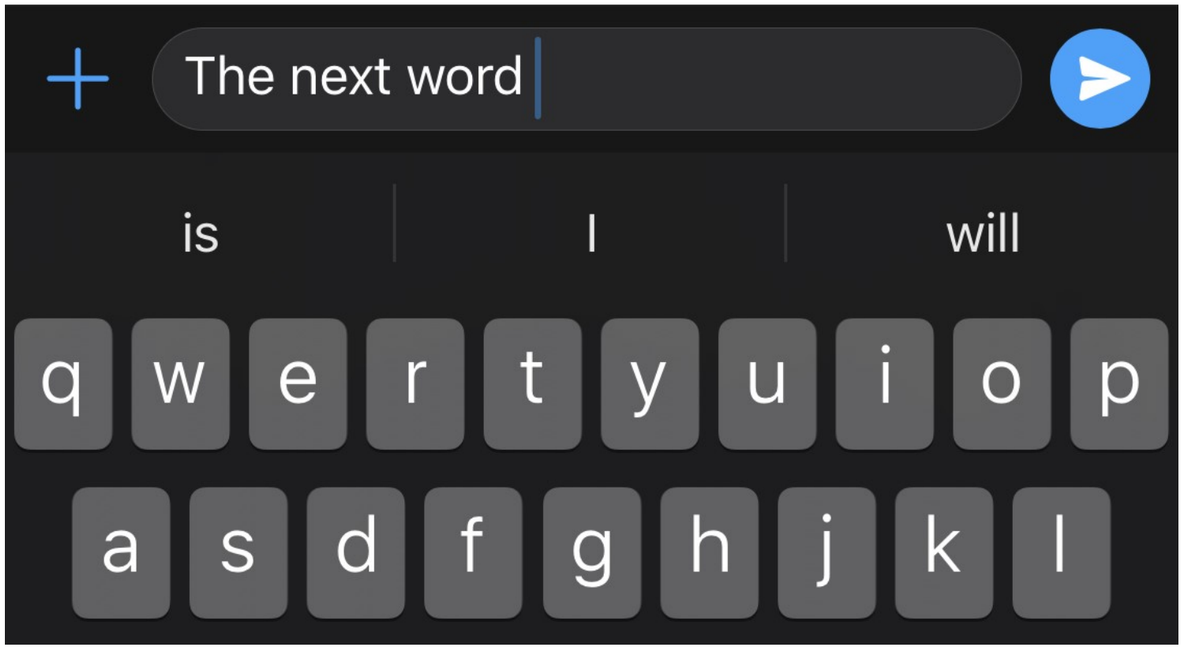
القسم الخامس : المعالجة المتقدمة للنصوص

الجزء الخامس :Language Modeling

==========

نتحدث الآن عن موضوع مهم , وهو : النماذج اللغوية , أو Language Modeling

و يقصد به : حساب احتمالية ان تكون هذه الجملة صحيحة كتكوين و لغة و تناسق مع بعضها البعض , فلو اردنا ترجمة جملة (I took my car to make my business) , فإن ترجمة (لقد قدت سيارتي لقضاء عملي) , أفضل من ترجمة (لقد اخذت سيارتي لصناعة تجارتي)



و هذه الاحتمالية بالغة الأهمية في العديد من التطبيقات , مثل :

- الترجمة الآلية , اختيار الكلمات المناسبة :
  - James is high or James is tall
- التصحيح التلقائي
  - I maked it yesterday == I make it yesterday or I made it yesterday
- معني الكلمة :
  - I saw her indeed , saw = (يري ام ينشر)
- التعرف علي الصوت :
  - اضربك بالألم او بالقلم
  - اخوك كل بالبحر

- Machine Translation:
  - P(**high** winds tonite) > P(**large** winds tonite)
- Spell Correction
  - The office is about fifteen **minuets** from my house
    - P(about fifteen **minutes** from) > P(about fifteen **minuets** from)
- Speech Recognition
  - P(I saw a van) >> P(eyes awe of an)
- + Summarization, question-answering, etc., etc.!!

_

و تقوم الفكرة علي حساب احتمالية تواجد كلمة معينة وراء كلمة أخري , فهل كلمة "ركب" و "سيارة" متناسبين اكثر ام "أخذ" و "سيارة"

كما اننا نقوم بحساب احتمالية تواجد كلمة معينة , عقب عدد من الكلمات الموجودة
"أحضرت الورق و الدفتر , وفتحته علي الصفحة المناسبة و أمسكت بـ " : القلم , الألم

- Goal: compute the probability of a sentence or sequence of words:

  $P(W) = P(w_1,w_2,w_3,w_4,w_5...w_n)$

- Related task: probability of an upcoming word:

  $P(w_5|w_1,w_2,w_3,w_4)$

- A model that computes either of these:

  $P(W)$ or $P(w_n|w_1,w_2...w_{n-1})$ is called a **language model**.

- Better: **the grammar** But **language model** or **LM** is standard

و لعمل هذه العملية الحسابية , نحتاج ان نستخدم مبدأ قاعدة السلسلة chain rule

و الذي ينص علي ان احتمالية تواجد عدد من الكلمات المتتالية = احتمالية تواجد الكلمة الأولي × احتمالية الكلمة الثانية مع معلومية الاولي × احتمالية تواجد الثالثة مع معلومية الاولي و الثانية . .. . و هكذا

More variables:

$$P(A,B,C,D) = P(A)P(B|A)P(C|A,B)P(D|A,B,C)$$

The Chain Rule in General

$$P(x_1,x_2,x_3,\ldots,x_n) = P(x_1)P(x_2|x_1)P(x_3|x_1,x_2)\ldots P(x_n|x_1,\ldots,x_{n-1})$$

لذا يتم تطبيق القانون هكذا :

P("its water is so transparent") =

P(its) × P(water|its) × P(is|its water)

× P(so|its water is) × P(transparent|its water is so)

ويتم استخدام ما يسمي قاعدة ماركوف , والتي تؤكد لنا أن احتمالية وجود كلمة معينة بناء علي جميع الكلمات السابقة , قريبة جدا من احتمالية تواجد هذه الكلمة بناء علي عدد قليل من الكلمات السابقة لها

فلو كان لدي جملة

I`m programmer & work in machine learning & I love python very ****

فاحتمالية تواجد الكلمة التالية much بناء علي كل الجملة السابقة , قريبة من احتمالية وجودها بناء علي كلمة او كلمتين فقط سابقة لها , و هذا يختصر الكثير من الوقت و المجهود

ويمكن اختصار القانون لـ :

$$P(w_i \mid w_1 w_2 \dots w_{i-1}) \approx P(w_i \mid w_{i-k} \dots w_{i-1})$$

-

و هذه هي فكرة الـ grams, حيث نقوم بتوقع الكلمة التالية بناء علي كلمات سابقة محددة

فهناك ال unigram حيث نتوقع كلمات عشوائية بلا رابط بينها , وبالطبع ستكون بلا معني , او ال Bigram ستكون كل كلمتين معا مناسبتين و لكن يظل السياق غامض , و قد تزيد للـ trigram او اكثر

و لكن المشكلة أن فكرة ال grams عاجزة عن بناء جمل كاملة حقيقية, لان الهيكل اللغوي اكثر تعقيدا من هذا , فجملة مثل :

أمس قابلت محمد و أخبرني انه عاني معاناة كبيرة مع المرض و انه كان يحاول الاتصال بي كثيرا و . . .

(انصرف , انصرفت , انصرفوا , انصرفن , ينصرف)

فاختيار كلمة " انصرف" , عائدة علي " محمد" علي الرغم انها بعيدة , وعلي الرغم من ان الكلمات السابقة لها ليست مرتبطة بها

_

و لحساب قيمة NGram يتم استخدام القانون التالي , و الذي يقوم بحساب عدد مرات وجود الكلمة الثانية عقب الأولي , مقسومة علي عدد مرات تواجد الكلمة الأولي في الملف كله

$$P(w_i \mid w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

$$P(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

فلو كان لدينا ثلاث جمل هكذا , يتم حساب هذه القيم بهذه الطريقة , مع الوضع في الاعتبار ان هناك بداية للجملة و نهاية لها

$$P(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

<s> I am Sam </s>
<s> Sam I am </s>
<s> I do not like green eggs and ham </s>

$P(\text{I} \mid \text{<s>}) = \frac{2}{3} = .67$ $P(\text{Sam} \mid \text{<s>}) = \frac{1}{3} = .33$ $P(\text{am} \mid \text{I}) = \frac{2}{3} = .67$
$P(\text{</s>} \mid \text{Sam}) = \frac{1}{2} = 0.5$ $P(\text{Sam} \mid \text{am}) = \frac{1}{2} = .5$ $P(\text{do} \mid \text{I}) = \frac{1}{3} = .33$

و بالتالي اذا كان لدينا عدد من الجمل , وقمنا بعمل هذا الحساب , فستكون القيم كالتالي :

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

و يبدو واضحا أن هناك ارقام كبيرة want + to و I + want , لأزواج الكلمات المنتشرة , وارقام قليلة : spend to

بعد هذا يتم عمل normalization اي يتم قسمة عدد مرات تواجد الكلمتين معا , علي عدد مرات تواجد الكلمة الاولي

Normalize by unigrams:

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

Result:

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

فتكون احتمالية تواجد هذه الجملة , هي احتمالية تواجد I عقب البداية , مضروبة في احتمالية تواجد want عقب I مضروبة في تواجد English عقب want و هكذا

P(<s> I want english food </s>) =

P(I|<s>)

× P(want|I)

× P(english|want)

× P(food|english)

× P(</s>|food)

= .000031

و نجد ان هناك ارقام معقولة (want Chinese) و ارقام اقل منها (want English) هذا بسبب انتشار المأكولات الصينية عن الانجليزية , و رقم كبير want to لان to مضافة لها غالبا , و رقم صفر to food و هذا بسبب عدم تواجدها في عينة التدريب لكنها ليست مستحيلة , و رقم مستحيل لغويا spend want

- P(english|want) = .0011
- P(chinese|want) = .0065
- P(to|want) = .66
- P(eat | to) = .28
- P(food | to) = 0
- P(want | spend) = 0
- P (i | <s>) = .25

و يتم عمل لوغاريتم الاحتمالات , وذلك لتجنب الحصول علي ارقام تقترب من الصفر (من ضرب ارقام صغيرة جدا) و لان الجمع اسرع من الضرب

$$\log(p_1 \times p_2 \times p_3 \times p_4) = \log p_1 + \log p_2 + \log p_3 + \log p_4$$

_

حسنا , كيف نقوم بعمل تقييم للموديل ؟

و هذا يمكن أن يتم عبر الطريقة التقليدية , وهي ان يتم تدريبه علي بيانات التدريب , ثم اختباره علي بيانات الاختبار

و قد يتم عمل اختبار له في مدي قدرته علي عمل تصحيح للـ spelling و غيرها من المهام و يتم حساب عدد المهام الناجحة و الفاشلة

لكن هذا الأمر احيانا يستغرق وقتا طويلا , لذا يتم استخدام تكنيك perplexity و يمكن ان نترجمها لكلمة **نسبة الخطأ**

و يتم هذا عبر استخدام الموديل الذي تم تدريبه علي جمل معروفة مسبقا , و تحديد النسبة التي قام هو بإعطائها للكلمة التي نعرف انها صحيحة

فلو كان لدينا جملة معروفة لدينا :

James went to school to take his exam

فنعطي للموديل الجملة بدون الكلمة الأخيرة , ونجعله يقوم بتوقع جميع الكلمات الممكنة للكلمة الاخيرة , ونبحث عن كلمة exam نري الاحتمالية التي اعطاها لها هذا الموديل

كما اننا يمكن البحث عن كلمات مشابهة بشرط ان يكون لها معني منطقي test , lesson و هكذا

The Shannon Game:

- How well can we predict the next word?

I always order pizza with cheese and ____

The 33rd President of the US was ____

I saw a ____

- Unigrams are terrible at this game. (Why?)

mushrooms 0.1

pepperoni 0.1

anchovies 0.01

....

fried rice 0.0001

....

and 1e-100

A better model of a text

- is one which assigns a higher probability to the word that actually occurs

و يكون لها قانون رياضي هو :

The best language model is one that best predicts an unseen test set

- Gives the highest P(sentence)

Perplexity is the inverse probability of the test set, normalized by the number of words:

$$PP(W) = P(w_1 w_2 ... w_N)^{-\frac{1}{N}}$$

$$= \sqrt[N]{\frac{1}{P(w_1 w_2 ... w_N)}}$$

Chain rule:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_1 \ldots w_{i-1})}}$$

For bigrams:

$$\mathrm{PP}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i | w_{i-1})}}$$

Minimizing perplexity is the same as maximizing probability

_

و هنا نقطع خطوات اكبر في موضوع تقييم الموديل

فلو كان لدينا موديل تم تدريبه بالكامل علي كل جمل شكسبير , و طلبنا منه عمل text generation اما بـ unigram or bigram or trigram or quadgram , فسيكون كالتالي :

Unigram

To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have
Every enter now severally so, let
Hill he late speaks; or! a more to leg less first you enter
Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like

Bigram

What means, sir. I confess she? then all sorts, he is trim, captain.
Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman?

Trigram

Sweet prince, Falstaff shall die. Harry of Monmouth's grave.
This shall forbid it should be branded, if renown made it empty.
Indeed the duke; and had a very good friend.
Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.

Quadrigram

King Henry.What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in;
Will you not tell me who I am?
It cannot be but so.
Indeed the short and the long. Marry, 'tis a noble Lepidus.

و نلحظ أن هناك اسلوب شكسبير , لكنه يفتقد للسياق السليم , خاصة في الموديلات الاولي , و السبب ان عدد كلمات جميع نصوص شكسبير تتكون من 300 الف زوج من الكلمات المتتالية , و اذا قمنا بحساب ال bigrams و التي ستكون مربع ضخم عدد صفوفه و اعمدته هي نفسها عدد الازواج , و ستكون القيم المربعة هي 800 مليون

و هذا معناه ان 99.96% من احتمالية ال bigrams لم تتواجد و ستكون بصفر , وهو ما سيمثل مشكلة كبيرة في الحساب

و نستنتج من هذا , أن فكرة الـ ngrams تنجح اذا ما تم استخدام الموديل في عمل كلمات علي داتا تم تدريبها عليه بالفعل , فلو تم تدريب الموديل علي شكسبير , فسيفشل الموديل تماما في استنتاج مقال صحفي بها . .

لكن المشكلة ان هذا عمليا بلا فائدة , فاذا كنا نريد استنتاج كلمات شخص معين , فهل علينا ان نستخلص داتا منه لنتدرب عليها ؟

و كأن المشكلة هي في الاصفار الناتجة عن bigrams لم تتواجد في بيانات التدريب , لكنها قد تأتي في بيانات الاختبار , والتي قطعا ستكون قيمة احتماليتها بصفر

فاذا تم تدريب الموديل علي الجمل اليسرى , فإذا جاءت أحد الجمل اليمني فستكون احتماليتها بصفر , علي الرغم من منطقية هذه الجمل , هي ما ستؤدي الي عدم امكانية حساب الـ perplexity لأنها سيتم قسمتها علي الصفر

Training set:
… denied the allegations
… denied the reports
… denied the claims
… denied the request

- Test set
  … denied the offer
  … denied the loan

P(“offer” | denied the) = 0

كيف يتم حل هذه المشكلة ؟

_

و يتم هذا عبر تكنيك add 1 و الذي يعني اضافة رقم 1 في المعادلة الاصلية , حتي تكون الكلمات التي تم يتم سردها في بيانات التدريب كـ bigram , يكون لها قيمة حتي لو ضئيلة

و كان التصور الاول (كلمات التدريب باحتمالية معينة و الكلمات المختفية بصفر) سيتحول للتصور الثاني (ان يتم استقطاع بعض الاحتمالية من الكلمات الموجودة , لحساب الكلمات المختفية من بيانات التدريب)

When we have sparse statistics:

P(w | denied the)

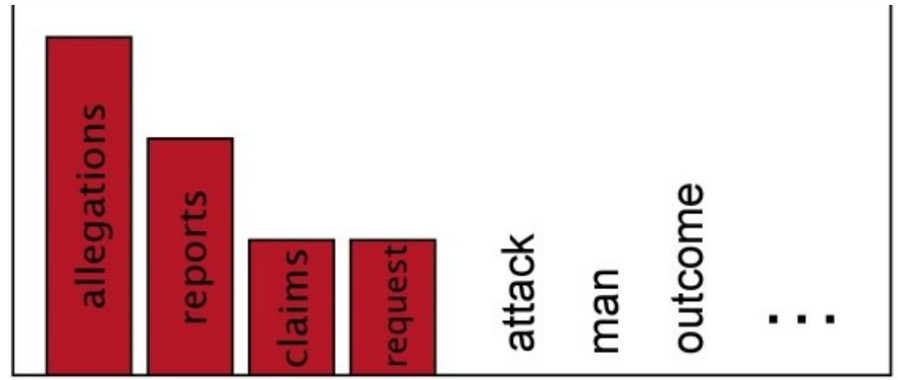
3 allegations

2 reports

1 claims

1 request

7 total



Steal probability mass to generalize better

P(w | denied the)

2.5 allegations

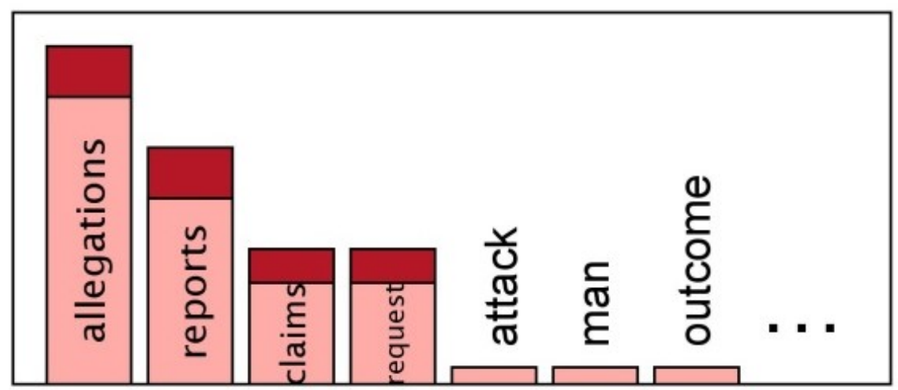
1.5 reports

0.5 claims

0.5 request

2 other

7 total



و يتم هذا عبر اضافة رقم 1 اثناء الحساب , حتي يكون للكلمات المختفية احتمالية ولو قليلة

Also called Laplace smoothing

Pretend we saw each word one more time than we did

Just add one to all the counts!

MLE estimate: $$P_{MLE}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

Add-1 estimate: $$P_{Add-1}(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i)+1}{c(w_{i-1})+V}$$

و بالتالي يكون جدول التكرار هكذا

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

و يكون جدول الاحتمالية هكذا

$$P^*(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)+1}{C(w_{n-1})+V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058