

Natural Language Processing

Matilde Marcolli

CS101: Mathematical and Computational Linguistics

Winter 2015

Reference

- C.D. Manning, H. Schütze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

- Setting based on **Probabilistic Linguistics**
- **Electronic Corpora**
 - Linguistic Data Consortium
 - European Language Resources Association
 - International Computer Archive of Modern English
 - Oxford Text Archive
 - Child Language Data Exchange System
- **Stemming**: stripping off affixes and word formation and extract stem of words from a word list
- **Markup**: syntactic structure is marked
- **Penn Treebank**: Lisp-like bracketing to mark binary tree structure of sentence
- **SGML** (Standard Generalized Markup Language): HTML is a type of SGML encoding; Text Encoding Initiative (TEI) encoding scheme suitable for marking parts of various texts, XML simplified form good for web applications

- **Grammatical Tagging:** automated tagging for categories (parts of speech: nouns, verbs,...)
- **Tag Sets:** American Brown Corpus (developed to tag the Lancaster–Oslo–Bergen corpus and British National Corpus)
- **Penn Treebank tag set:** most widely used in computational setting (simplified version of previous)
- **rule:** least marked category is used as default whenever a word cannot be placed in any other more precise subcategory with additional markings
- **Example:** “Adjectives” used if cannot further place into “comparatives, superlatives, numerals,...”
- available tag sets are very different (some coarser, some more refined)

- **good criterion for tag sets**: want to use tag of parts of speech that “best predicts” the behavior of nearby words
- some tags may be too coarse to be good predictors
- Example: adverb *not* has very different distribution than other adverbs, but if tagged in the same way the distinction is lost
- Example: should one use same tag for auxiliaries and for other verbs? (Penn Treebank does)
- correctly identifying proper names is an issue

- at both syntactic and semantic level **punctuation** matters...
 - *the panda bear eats shoots and leaves*
 - *the panda bear eats, shoots, and leaves*
- **Collocations**: a collection of two or more words with **limited compositionality**
- **compositionality**: meaning of whole expression deduced from meaning of parts
- Example: an idiom is not compositional
I heard it through the grapevine
- Example: a combination of adjective+noun which cannot be replaced by other synonyms
strong tea (cannot replace “strong” with other words referring to “physical strength” in the same context)
- Example: technical terms are also examples of collocation

- Collocations are identified by **frequency based search** (for combination of words)
- **Collocation window**: catch combinations of words that are not contiguous in a sentence
- Example: identify “knock – door” as a collocation (cannot replace “knock” with another term like “hit” or “beat”, it would not be correct) requires a window of several words
they knocked repeatedly at the metal front door
- the collocation in such cases can be identified by computing the **mean and variance** of the signed distance between the two words over the corpus
- collocations = pairs with low deviation (if two unrelated words, variance would be higher)

- still high frequency and low variance can happen by accident: need to **test hypothesis**
- **null hypothesis** (no association) and compute probability of event under this hypothesis, reject if too low (below a given significance level)
- under null hypothesis (independence): $\mathbb{P}(w_1, w_2) = \mathbb{P}(w_1) \cdot \mathbb{P}(w_2)$
- **t-test** for collocations: $\mathbb{P}_f(w_1, w_2)$ = frequency, number of occurrences normalized by corpus size N

$$t = \frac{\mathbb{P}_f(w_1, w_2) - \mathbb{P}(w_1) \cdot \mathbb{P}(w_2)}{\sqrt{\sigma^2/N}}$$

σ = variance

- if t -value is not larger than critical value (for a desired “confidence level”) then cannot reject null hypothesis

- use for **disambiguation** for synonyms
- Example: *strong* and *powerful* can sometime be used as synonyms but not always
- look for other words that form a collocation with one but not with the other (as a way to separate their meaning)
- Examples: *powerful computer*; *strong support*
- w_1 and w_2 the words being compared; w another word

$$t \sim \frac{\mathbb{P}_f(w, w_1) - \mathbb{P}_f(w, w_2)}{\sqrt{\mathbb{P}_f(w, w_1) + \mathbb{P}_f(w, w_2)}}$$

comparison of the means of two independent normal populations

- t-test assumes probabilities normally distributed
- more general test: χ^2 -test
- form a 2×2 matrix of frequencies $\mathbb{P}(v_1, v_2)$

$$(O_{ij}) = \begin{pmatrix} \mathbb{P}_f(v_1 = w_1, v_2 = w_2) & \mathbb{P}_f(v_1 = w_1, v_2 \neq w_2) \\ \mathbb{P}_f(v_1 \neq w_1, v_2 = w_2) & \mathbb{P}_f(v_1 \neq w_1, v_2 \neq w_2) \end{pmatrix}$$

then compute

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

with expected values E_{ij} under null hypothesis

- can apply to other sizes than 2×2
- use χ^2 as **measure of corpus similarity** on an $N \times 2$ matrix
 N = list of words, 2 = two corpora being compared, record frequencies of words in the two corpora

- use χ^2 for **identification of translation pairs** in **aligned corpora**
- alignment: know which sentence matches which sentence in the overall text
- count occurrences of words in matching sentences (by 2×2 -matrix: frequency of both occurring, of one but not other, of neither)
- high value of χ^2 (with respect to E_{ij} of independence null-hypothesis) reveals a matching translation pair with high confidence

- **Likelihood ratio** another method for hypothesis testing
- compare independence hypothesis formulated as

$$\mathbb{P}(w_1 | w_2) = \mathbb{P}(w_1 | \neg w_2) = p$$

with dependence case

$$p_1 = \mathbb{P}(w_1 | w_2) \neq \mathbb{P}(w_1 | \neg w_2) = p_2$$

- occurrences counting: c_1 , c_2 , c_{12} for w_1 , w_2 and both
- **Example**: assume a **binomial distribution**

$$b(k; n, x) = \binom{n}{k} x^k (1-x)^{n-k}$$

- then c_{12} out of c_1 bigrams are $w_1 w_2$:
 $b(c_{12}; c_1, p)$ in first case; $b(c_{12}; c_1, p_1)$ in second case
- $c_2 - c_{12}$ out of $N - c_1$ bigrams are $\neq w_1 w_2$
 $b(c_2 - c_{12}; N - c_1, p)$ in first case; $b(c_2 - c_{12}; N - c_1, p_2)$ in second

- **likelihood** of independence hypothesis is product

$$\mathcal{L}_1 = b(c_{12}; c_1, p) \cdot b(c_2 - c_{12}; N - c_1, p);$$

likelihood of dependence hypothesis is product

$$\mathcal{L}_2 = b(c_{12}; c_1, p_1) \cdot b(c_2 - c_{12}; N - c_1, p_2)$$

- **likelihood ratio**

$$\lambda = \frac{\mathcal{L}_1}{\mathcal{L}_2}$$

or in logarithmic form $\log \lambda$

- this test works better than t-test and χ^2 -test for **sparse data**

- **Information theoretic** test for collocations:
pointwise mutual information

$$\mathcal{I}(X, Y) = \log_2 \frac{\mathbb{P}(X, Y)}{\mathbb{P}(X)\mathbb{P}(Y)} = \log_2 \frac{\mathbb{P}(X | Y)}{\mathbb{P}(X)} = \log_2 \frac{\mathbb{P}(Y | X)}{\mathbb{P}(Y)}$$

- how much more information about the occurrence of a word w_2 if know occurrence of word w_1
- case of **total dependence**

$$\mathcal{I}(X, Y) = \log \frac{\mathbb{P}(X, Y)}{\mathbb{P}(X)\mathbb{P}(Y)} = \log \frac{\mathbb{P}(X)}{\mathbb{P}(X)\mathbb{P}(Y)} = -\log \mathbb{P}(Y)$$

- case of **perfect independence**

$$\mathcal{I}(X, Y) = \log \frac{\mathbb{P}(X, Y)}{\mathbb{P}(X)\mathbb{P}(Y)} = \log \frac{\mathbb{P}(X)\mathbb{P}(Y)}{\mathbb{P}(X)\mathbb{P}(Y)} = 0$$

values close to zero: approximate independence

- **Caveat:** this measure of increased information does not explain nature of the relation
- Example: *Chambre des Communes* in French Canadian translates *House of Commons* so in a corpus of text from Canadian parliament would find high $I(X, Y)$ for both pairs (*house, chambre*) and (*house, communes*)
- χ^2 -test identifies correctly which pair is better match (because of other unmatched occurrences)
- but $\mathcal{I}(X, Y)$ can give higher value of second

$$\mathcal{I}(X, Y) = \frac{\mathbb{P}(X, Y)}{\mathbb{P}(Y | X) + \mathbb{P}(Y | \neg X)} \cdot \frac{1}{\mathbb{P}(X)}$$

depending on the entries $\mathbb{P}(Y | X)$ and $\mathbb{P}(Y | \neg X)$ of test matrix (for same X and for the two choices of Y being compared)

- Note: all these test *not* very good for **low frequency** cases (rare digrams)

- the above is **pointwise** mutual information; while usual information theoretic **mutual information** is averaged (expectation value)

$$\begin{aligned} \mathbb{I}(X, Y) &= \mathbb{E}_{\mathbb{P}(x,y)} (\mathcal{I}(x, y)) = \mathbb{E}_{\mathbb{P}(x,y)} \left(\log \frac{\mathbb{P}(x, y)}{\mathbb{P}(x)\mathbb{P}(y)} \right) \\ &= \sum_{x \in X, y \in Y} \mathbb{P}(x, y) \log \frac{\mathbb{P}(x, y)}{\mathbb{P}(x)\mathbb{P}(y)} \end{aligned}$$

- the **pointwise** version measures reduction of uncertainty about occurrence of one word knowing about occurrence of the other
- the **averaged** version has similar meaning, but only measures it in an averaged sense

More about word disambiguation

- when the same word can have several different meanings, find a way to attribute the appropriate one in the context of the sentence
- also same word (in English) can often be used as a noun or a verb: problem of tagging correctly as part of speech
- **supervised** versus **unsupervised** learning: know status (label) of each item used for training or not
- large scale training sets can be created by the use of **pseudowords** (artificial conflation of two different words to create a disambiguation problem in a text)
- disambiguation algorithms use a **limited representation of context**: e.g. up to 3 words on each side of the ambiguous word

Supervised disambiguation

- a disambiguated corpus available for training
- **Bayesian classification** method: Bayesian decision rule choose s' if $\mathbb{P}(s' | c) > \mathbb{P}(s | c)$ for all other $s \neq s'$ where

$$\mathbb{P}(s | c) = \frac{\mathbb{P}(c | s)}{\mathbb{P}(c)} \mathbb{P}(s)$$

prior probability $\mathbb{P}(s)$ (not knowing anything about context), the other factor is evidence about the context

$$s' = \arg \max_s \mathbb{P}(s | c) = \arg \max_s \frac{\mathbb{P}(c | s)}{\mathbb{P}(c)} \mathbb{P}(s)$$

$$= \arg \max_s \mathbb{P}(c | s) \mathbb{P}(s) = \arg \max_s (\log \mathbb{P}(c | s) + \log \mathbb{P}(s))$$

assign meaning s to ambiguous word w

Naive Bayes classifier assumption

$$\mathbb{P}(c | s) = \mathbb{P}(\{v_j : v_j \in c\} | s) = \prod_j \mathbb{P}(v_j | s)$$

v_j = words that occur in the context c of the ambiguous word

$$s' = \arg \max_s \left(\log \mathbb{P}(s) + \sum_j \log \mathbb{P}(v_j | s) \right)$$

- the $\mathbb{P}(v_j | s)$ and $\mathbb{P}(s)$ computed from labelled training corpus

$$\mathbb{P}(v_j | s) = \frac{C(v_j, s)}{\sum_r C(v_r, s)}, \quad \mathbb{P}(s) = \frac{C(s)}{C(w)}$$

$C(v, s)$ = counting of occurrences of pair

- naive assumption does not work when there are strong conditional dependencies between them

- **Information theoretic** supervised disambiguation instead of using all words in a context window for disambiguation, tries to identify a single feature that forces disambiguation
- Example: want to distinguish the fact that *prendre une mesure* translates as *to take a measurement* while *prendre une décision* translates as *to make a decision*
- use (averaged) **mutual information**

$$I(X, Y) = \sum_{x \in X, y \in Y} \mathbb{P}(x, y) \log \frac{\mathbb{P}(x, y)}{\mathbb{P}(x)\mathbb{P}(y)}$$

- **Flip-flop algorithm:** linear-time algorithm computing best partition for given values of an indicator (requires labelled training set: supervised)
- $P = \{t_1, \dots, t_m\}$ set of possible translations of ambiguous word;
 $Q = \{x_1, \dots, x_m\}$ set of possible values of indicator;
- start with a random partition $P = P_1 \cup P_2$
- find partition $Q = Q_1 \cup Q_2$ maximizing $\mathbb{I}(P, Q)$
- find partition $P = P_1 \cup P_2$ maximizing $\mathbb{I}(P, Q)$ given partition of Q , etc.
- with each iteration $\mathbb{I}(P, Q)$ non-decreasing: stop when stationary

Dictionary based disambiguation

- D_1, \dots, D_k dictionary definitions of the meanings s_1, \dots, s_k of a word w
- list of words occurring in these definitions
- $v_j =$ words occurring in a context window c around w
- $E_{v_j} =$ vocabulary definition of v_j (list of words for all its meanings)
- **score:** $\sigma(s_k) = \#(D_k \cap \cup_j E_{v_j})$
count overlap in dictionary definitions
- choose meaning that maximizes: $s = \arg \max_k \sigma(s_k)$

Unsupervised disambiguation

- “sense tagging” cannot be completely unsupervised (assigning meaning to occurrences), but “sense discrimination” can
- cluster contexts of an ambiguous words into a number of (unlabeled) groups
- **Context-group discrimination** algorithm:
parameters $\mathbb{P}(v_j | s_k)$ and $\mathbb{P}(s_k)$
- initialize parameters randomly
- compute likelihood of corpus C given model μ

$$\mathbb{L}(C | \mu) = \sum_i \log \sum_k \mathbb{P}(c_i | s_k) \mathbb{P}(s_k)$$

c_i = individual contexts within corpus

$$\mathbb{P}(c_i | s_k) = \prod_j \mathbb{P}(v_j | s_k)^{C(v_j \in c_i)}$$

naive Bayes assumption

- estimate posterior probability

$$h_{ik} = \frac{\mathbb{P}(s_k)\mathbb{P}(c_i | s_k)}{\sum_k \mathbb{P}(s_k)\mathbb{P}(c_i | s_k)}$$

- re-estimate parameters

$$\mathbb{P}(v_j | s_k) = \frac{\sum_i \mathbb{C}(v_j \in c_i)h_{ik}}{\sum_j \sum_i \mathbb{C}(v_j \in c_i)h_{ik}}$$

$$\mathbb{P}(s_k) = \frac{\sum_i h_{ik}}{\sum_{ik} h_{ik}}$$

- stop when likelihood $\mathbb{L}(C | \mu)$ no longer increasing significantly
- choose $s' = \arg \max \left(\log \mathbb{P}(s_k) + \sum_j \mathbb{P}(v_j | s_k) \right)$
- can distinguish usage-types more fine grained than listed in dictionaries

Machine Translation

- increasingly refined levels of machine translation
- **Word-by-word** translations: riddled with lexical ambiguities, messes up word order
- **Syntactic parsing and syntactic transfer**: restores correct word order; ambiguities remain in ambiguities of parsing trees
- **Semantic representation and semantic transfer**: correcting for syntactic parse trees ambiguities
- **Interlingua**: knowledge representation formalism, instead of translation systems for each pair of languages $O(n^2)$ translate between each and interlingua level $O(n)$... problem: extremely difficult to design an efficient comprehensive representation formalism

Text Alignment first step for Machine Translation

- **matching sentences** in two parallel texts in different languages
- order of subordinate sentences can vary in different languages: need to match which sentence pairs with which
- good **training corpora** are legislation documents in bilingual or multilingual countries (Canada, Switzerland, Belgium,...) large supply of documents and accurately translated for reason of legal requirements

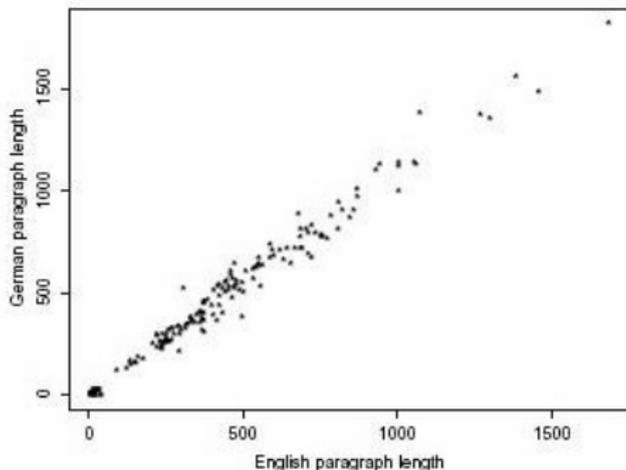
Length-based methods: want alignment A with highest probability

$$\arg \max_A \mathbb{P}(A | S, T)$$

given parallel texts S, T ; assume $\mathbb{P}(A | S, T) \sim \prod_k \mathbb{P}(B_k)$ sequence B_k of **alignment beads** statistically independent

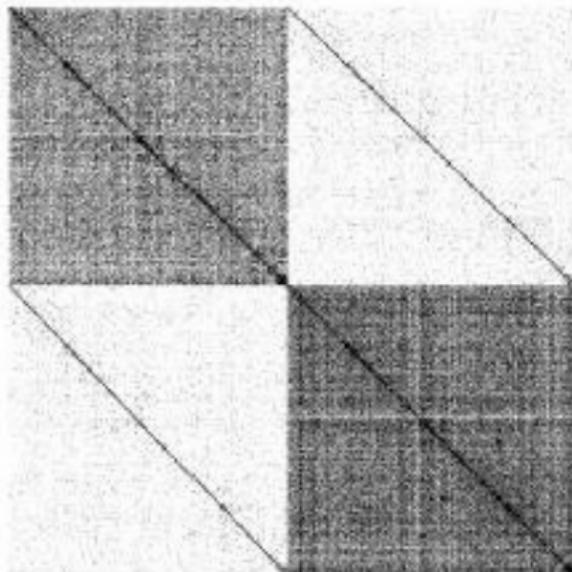
- estimate $\mathbb{P}(B_k)$ by comparing lengths of texts in B_k
- assign a **cost function** to each choice of alignment B_k and minimize (e.g. based on square of differences of lengths of paragraphs/sentences)
- this method breaks down easily on noisy data like optical character recognition (OCR)

Paragraph Lengths are Highly Correlated



Offset alignment methods by signal processing techniques:

- induce alignment based on **cognate words**: works best for languages in same family with many cognate words, but can work in other cases too based on proper nouns, numbers, and other such strings of characters
- **dot-plot**: concatenate both text and make big matrix: place dot at (x, y) if there is a match between positions x and y in the concatenated text... **bitext map** in off-diagonal quadrants
- can refine with counting positions in text where a word occurs and a vector of distances between subsequent positions, can compare with a candidate match: if distances too different discard... works better when punctuation may be lost (OCR) and languages are not in same family



- W.A.Gale, K.W.Church, *A Program for Aligning Sentences in Bilingual Corpora*, Computational Linguistics (1993) 19(3), 75–102

Word Alignment: next step after sentences matched, done from a bitext using association methods like χ^2 -test

Statistical Machine Translation: based on word alignment

$$\mathbb{P}(S_f | S_e) = \frac{1}{Z} \sum_{a_1 \dots a_m, =0}^{\ell} \prod_{j=1}^m \mathbb{P}(w_{f,j} | w_{e,a_j})$$

S_e, S_f English and French sentences, $\ell = \ell(S_e)$, $m = \ell(S_f)$ lengths in number of words, $w_{f,j}$ = j -th word of S_f , a_j = position of S_e that $w_{f,j}$ is aligned to, with word w_{e,a_j} , and $\mathbb{P}(w_{f,j} | w_{e,a_j})$ = translation probability (of finding $w_{f,j}$ as a translation of w_{e,a_j})

- sum counts all possible alignments ($a_j = 0$ if word with no matching translation on other side)

Decoder

$$\begin{aligned} S'_e &= \arg \max_{S_e} \mathbb{P}(S_e | S_f) = \arg \max_{S_e} \frac{\mathbb{P}(S_e)\mathbb{P}(S_f | S_e)}{\mathbb{P}(S_f)} \\ &= \arg \max_{S_e} \mathbb{P}(S_e)\mathbb{P}(S_f | S_e) \end{aligned}$$

need a **search algorithm** (in principle searching over an infinite space: need to reduce to a manageable search)

- **Stack search**: build sentence S'_e incrementally, keeping track of partial translation hypotheses, extend hypotheses (more words) and prune the stack of least likely extended hypotheses...

Translation Probabilities: estimate the $\mathbb{P}(w_f | w_e)$

- initialize $\mathbb{P}(w_f | w_e)$ randomly
- count occurrences in all pairs (S_e, S_f) of aligned sentences containing w_e and w_f

$$z_{w_f, w_e} = \sum_{(S_e, S_f)} \mathbb{P}(w_f | w_e)$$

- re-estimate probabilities

$$\mathbb{P}(w_f | w_e) = \frac{z_{w_f, w_e}}{\sum_v z_{v, w_e}}$$

Ethical issues in machine translation

- try Google Translate between English and German:
 - *math teacher* → *Mathlehrer*
 - *cooking teacher* → *Kochlehrerin*
 - *physics teacher* → *Physiklehrer*
 - *French teacher* → *Französisch-Lehrerin*
- because machine translators are primed on counting occurrences in available text corpora, they carry along with them whatever **cultural bias** exists in the society that produces those texts at the time in which they are produced
- is it possible to construct **unbiased** machine translators that avoid this kind of problem? It certainly would be desirable...

Information Retrieval

- developing algorithms for retrieving information from repositories of written texts
- **inverted index**: lists for each word all the documents that contain that word
- with additional **position information** of all the occurrences
- allows for search for **phrases** (combinations of words)
- **stop list** eliminates words not useful for search (too general such as “and”)
- **stemming** of words

Ranking of relevant documents is the main problem

- **Probability ranking principle:** ranking documents by decreasing probability of relevance is optimal
- view retrieval as a “greedy search” that aims at the most valuable document with highest \mathbb{P}
- method assumes that documents are independent

Vector space model:

- it uses spatial proximity for semantic proximity
- very high-dimensional space with one dimension for each word in the document collection
- measure of vector similarity

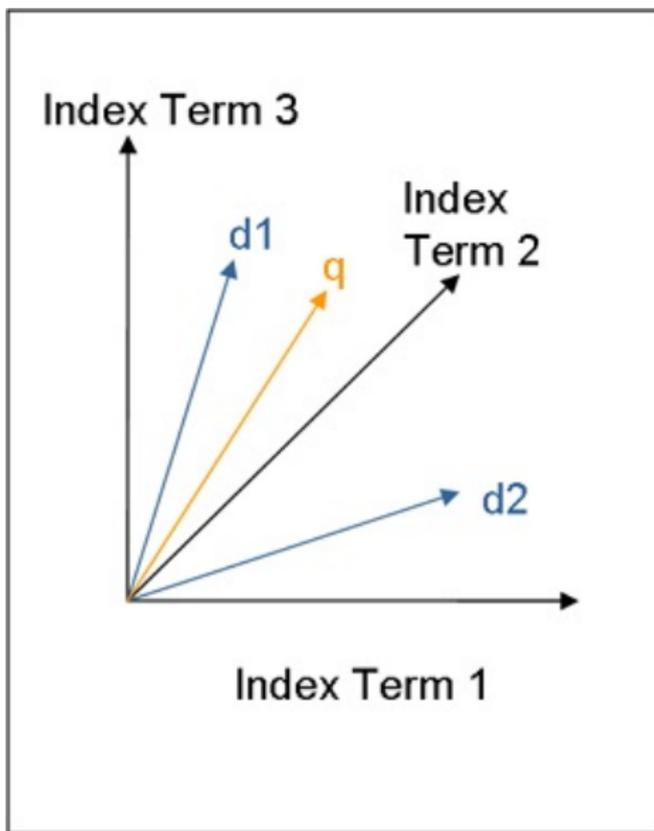
$$\frac{\langle v, w \rangle}{\|v\| \cdot \|w\|} = \cos(\theta(v, w))$$

- **Term weighting**: weight gives coordinates d_w to document-vectors $d = (d_w)$ in the vector space model
 - **term frequency**: $tf_{i,j}$ = occurrences of word w_i in document d_j
 - **document frequency**: df_i = number of documents in the collection in which w_i occurs
 - **collection frequency**: cf_i = total number of occurrences of w_i in the collection
- **weight**: if $tf_{i,j} \neq 0$

$$\text{weight}(i, j) = (1 + \log(tf_{i,j})) \log \frac{N}{df_i}$$

and zero otherwise, N = number of documents

- note **inverse document frequency**: max weight to words occurring in only one document, zero weight to words occurring in all



Term Distribution Models: alternative to weights, use model of occurrences of words (Zipf's law, or a distribution like Poisson) and decide how informative a word is

- Poisson distribution typical for events over units of a fixed size

$$p(k; \lambda) = e^{-\lambda} \frac{\lambda^k}{k!}$$

mean and variance both equal to λ

- use with $\lambda = cf_i/N$ average number of occurrences per document
- observed deviation from a Poisson distribution can signal “clustering” of term occurrences in only certain types of documents in the collection

Other probability distributions

- two-Poisson

$$p(k; \pi, \lambda_1, \lambda_2) = \pi e^{-\lambda_1} \frac{\lambda_1^k}{k!} + (1 - \pi) e^{-\lambda_2} \frac{\lambda_2^k}{k!}$$

separates documents containing w into a class with low number of occurrences and one with high

- K-mixture

$$p(k) = (1 - \alpha) \delta_{k,0} + \frac{\alpha}{\beta + 1} \left(\frac{\beta}{\beta + 1} \right)^k$$

α, β parameters can be fit using observed mean $\lambda = cf/N$ and observed inverse document frequency N/df