UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

TITOL

SUBTITOL

Autor

Data defensa
Director
Department director

MASTER IN INNOVATION AND RESEARCH IN INFORMATICS
Especialitat

Abstract

aaa

Contents

Chapter 1

Introduction

Aquesta part es important, ha de quedar clara

- marc motivació
  - model existent pot explicar mes lleis
- patrons llengues

1.1 State of the art / background / ...

- ...

1.2 Objectius generals

- quines lleis linguistiques es reprodueixen
- ...

1.3 intro model

- teoria informació
  - entropi
  - informació mutua
  - formules
- graf bipartit
  - conceptes del graf bipartit
  - sense matematiques
  - no hi ha arestes entre vertexos del mateix grup

  - grups: paraules i significats

- parametre phi
- intro primer model
  - conceptes del primer model
  - sense matematiques
  - probabilitat d'una paraula depen del nombre de significats
  - probabilitat d'un significat depen del nombre de paraules
- intro segon model
  - conceptes del segon model
  - sense matematiques
  - probabilitat de significats depen de probabilitat a priori
- optimització
  - omega a partir de formules teoria informació
  - objectiu: minimitzar omega
- dinamic vs estatic
  - explicar perque es deriven equacions dinamiques
  - explicar que representen (el que passa quan hi ha un canvi en A)

1.4 Objectius

- reproduir resultats antics
- crear eina codi obert
  - pot reproduir resultats antics, resolent problemes de replicabilitat
  - pot crear nous resultats amb varis parametres
- optimització: minims locals?
- ...

1.5 Hipòtesis

- article vitevith dona una relacio entre num significats
- lleis linguistiques surten de la optimitzacio
- negligim interacció social (unlike e.g. naming game)
- la optimitzacio no arriba necessariament a un optim global

1.6 problemes

1.6.1 llinguistica quantitativa

- que prediuen els models actuals

1.6.2 computacional

- calcular omega eficientment
- asegurar correctesa (tests)
- condicio aturada optimitzacio

1.6.3 calcul dinamic

- complexitat matematica
- error numeric

1.6.4 outline of the thesis

- cadascuna de les parts de la tesis
- mencionar el que al final s'explica a la discussió

En algun lloc de la intro: Taula de resum dels models.

- Columnes: Ramon typing, Simon's model, Model 2003 amb phi = 0, Model 2005 amb phi = 0, Model 2003 amb phi = 1, Model 2005 amb phi = 1.
- Rengles: Zipf rank-frequency law, Zipf's law of meaning distribution, Zipf's law of the age of words, vocabulary learning bias.
- Cel·les: Yes, No, Aclariments.

Chapter 2

Model

This chapter covers the details about the two models previously introduced in Chapter 1, where a high level introduction to both the proportional model and the *a priori* model was given. It follows a top down approach. General concepts from both types of model are outlined first and the specific details from each model are given afterwards.

As a reminder of the previous chapter a very brief definition of the two models follows. Both models are quite similar, they represent a way to connect words with meanings. Both meanings and words are given probabilities of being used, from which information theoretic measures are derived, which are ultimately used to obtain a cost function. The high level differentiating trait is the way in which the probability of a meaning is obtained. In the first studied model (proportional model), the probabilities of both words and meanings are proportional to the number of connections. In the second model (*a priori* model), the probabilities of meanings come from a distribution of *a priori* probabilities.

The remainder of this chapter is divided into two sections. Section 2.1 deals with the mathematical definitions of the models and it is more theoretical in nature. Section 2.2 outlines the computational aspects of the two models and it is more practical and implementation oriented. This chapter does not deal with any actual details of the implementation of the model. These details can be found in Chapter 3.

Note that Chapter 1 offers a high level revision of the graph theory and information theory concepts that will appear during the rest of this chapter.

2.1 Mathematical aspect

This section covers the mathematical definition of the model. Section 2.1.1 introduces concepts common to both models. These concepts are then extended for either the proportional model or the *a priori* model in Sections 2.1.2 and 2.1.3 respectively.

2.1.1 Common concepts

In this section, mathematical definitions and notation common to both models is given. They will be used throughout the remainder of the thesis.

Both models studied in this thesis are based on the idea of a bipartite graph $G_{n,m}$. Bipartite graphs are comprised of two sets of elements and edges can only appear between an element of one set and an element of the other.

Our two sets are S and R. S is a set of size n containing all words. The notation s_i is used to refer to some element i of the set S. R is a set of size m containing all meanings. The notation r_j is used to refer to some element j of the set R.

The bipartite graph is represented using the adjacency matrix $A_{n,m}$ (or simply A) in most cases. $A_{n,m}$ is a $n \times m$ binary matrix representing whether an edge exists or not in the graph. Mathematically, each element of the matrix A is defined as

$$a_{i,j} = \begin{cases} 1 & \text{if there exists an edge between } s_i \in S \text{ and } r_j \in R \\ 0 & \text{otherwise.} \end{cases}$$

In some cases it is more convenient to represent the bipartite graph as the set E of all edges,

$$E = \{\, (s_i, r_j) \mid \text{there exists an edge between } s_i \in S \text{ and } r_j \in R \,\}.$$

For brevity, the tuple (s_i, r_j) is sometimes replaced by the simpler form (i, j).

A notation is defined for the degree of the vertices in the graph. The word s_i has degree μ_i, while the meaning r_j has degree ω_j. Mathematically, μ and ω are defined

$$\mu_i = \sum_{j=1}^{m} a_{i,j} \tag{2.1}$$

and

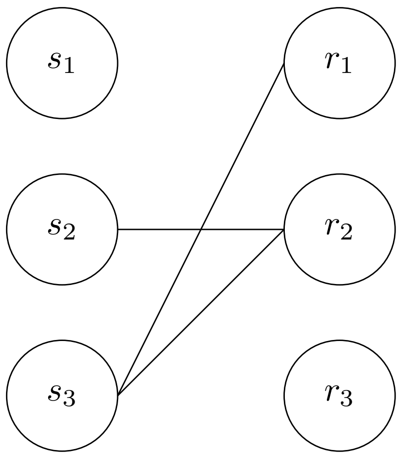
$$\omega_j = \sum_{i=1}^{n} a_{i,j}. \tag{2.2}$$

There is an additional parameter of the models, ϕ, which is not directly related to the bipartite graph. This parameter appears in both models and it is used to "emphasize" the effect of the vertex degrees in the calculations. As explained in previous sections, this parameter is a new addition to the models originally presented in [2] and [3].

Figure 2.1 displays an example of a simple $G_{3,3}$ graph. A graphical representation is included, as well as the corresponding values of the mathematical concepts discussed in this section up to here.

With the parameter ϕ, the definitions of μ and ω can be generalized. μ_ϕ and ω_ϕ are defined as

$$\mu_{\phi,i} = \sum_{j=1}^{m} a_{i,j}\omega_j^{\phi} \tag{2.3}$$



$$S = \{\, s_1, s_2, s_3 \,\} \qquad \mu_1 = 0$$
$$R = \{\, r_1, r_2, r_3 \,\} \qquad \mu_2 = 1$$
$$\mu_3 = 2$$

$$\omega_1 = 1$$
$$\omega_2 = 2$$
$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \qquad \omega_3 = 0$$

$$E = \{\, (s_2, r_2), (s_3, r_1), (s_3, r_2) \,\}$$

Figure 2.1: A $G_{3,3}$ bipartite graph with the corresponding adjacency matrix A, set of edges E and vertex degrees μ and ω for each vertex.

and

$$\omega_{\phi,i} = \sum_{i=1}^{n} a_{i,j} \mu_i^{\phi}. \tag{2.4}$$

It can be seen that when $\phi = 0$ Equation (2.3) becomes Equation (2.1) and Equation (2.4) becomes Equation (2.2). In other words, $\mu_{0,i} = \mu_i$ and $\omega_{0,j} = \omega_j$

2.1.2 The proportional model

This model is defined by the joint probability of a word s_i and a meaning r_j. This probability must be proportional to the product of the degrees of s_i and r_j and it must be zero if there is no edge between s_i and r_j. Mathematically,

$$p(s_i, r_j) \propto a_{i,j} (\mu_i \omega_j)^{\phi},$$

where the parameter ϕ is used to control the effect of the vertex degrees. As previously noted, the original model from [2] is equivalent to this one but with $\phi = 0$ and so, the vertex degrees have no effect in the joint probability. no ho tinc gaire clar... desde on "comencen" les equacions del primer model?

From this definition, equations for the marginal probabilities are derived, which in turn are used to obtain the information theory expressions in the cost function Ω (Equation (**??**)). From these expressions, dynamic equations are derived to obtain the change experienced by the entropies after a single mutation has occurred on the adjacency matrix A.

Joint Probability

Adding a normalizing factor M_ϕ, the joint probability becomes

$$p(s_i, r_j) = \frac{1}{M_\phi} a_{i,j} (\mu_i \omega_j)^\phi. \tag{2.5}$$

This normalizing factor is obtained by applying the definition of probability

$$\sum_{i=1}^{n} \sum_{j=1}^{m} p(s_i, r_j) = 1.$$

Then,

$$M_\phi = \sum_{i=1}^{n} \sum_{j=1}^{m} a_{i,j} (\mu_i \omega_j)^\phi$$

or equivalently

$$M_\phi = \sum_{(i,j)\in E}^{n} (\mu_i \omega_j)^\phi. \tag{2.6}$$

Marginal Probabilities

Formulas for the marginal probabilities $p(s_i)$ and $p(r_j)$ are derived from the joint probability using the general formula

$$p(x) = \sum_{y \in Y} p(x, y). \tag{2.7}$$

Applying Equation (2.7),

$$p(s_i) = \frac{\mu_i^\phi}{M_\phi} \sum_{j=1}^{m} a_{i,j} \omega_j^\phi.$$

Recall Equation (2.3),

$$p(s_i) = \frac{\mu_i^\phi \mu_{\phi,i}}{M_\phi}. \tag{2.8}$$

Symmetrically applying Equation (2.7) to obtain $p(r_j)$ and applying Equation (2.4) instead we obtain

$$p(r_j) = \frac{\omega_j^\phi \omega_{\phi,j}}{M_\phi}. \tag{2.9}$$

Entropies

$H(S, R)$ is obtained by applying the joint probability found in Equation (2.5) to Equation (**??**),

$$H(S, R) = -\sum_{i=1}^{n} \sum_{j=1}^{m} \frac{1}{M_\phi} a_{i,j} (\mu_i \omega_j)^\phi \log \left(a_{i,j} (\mu_i \omega_j)^\phi \right). \tag{2.10}$$

This expression can be refined taking advantage of the equality

$$-\sum_i \frac{x_i}{T}\log\frac{x_i}{T} = \log T - \frac{1}{T}\sum_i x_i \log x_i \tag{2.11}$$

which holds as long as

$$T = \sum_i x_i.$$

The full derivation of Equation (2.11) can be found in Appendix B.1.

Applying Equation (2.11) to Equation (2.10) we obtain

$$H(S,R) = \log M_\phi - \frac{\phi}{M_\phi}\sum_{i=1}^{n}\sum_{j=1}^{m} a_{i,j}\left((\mu_i\omega_j)^\phi \log a_{i,j}\mu_i\omega_j\right)$$

using E in the summation and adopting the convention that $0\log 0 = 0$

$$H(S,R) = \log M_\phi - \frac{\phi}{M_\phi}\sum_{(i,j)\in E}(\mu_i\omega_j)^\phi \log \mu_i\omega_j \tag{2.12}$$

is reached.

The marginal word entropy is found applying the marginal probability in Equation (2.8) to the definition of $H(S)$ in Equation (**??**)

$$H(S) = -\sum_{i=1}^{n}\frac{\mu_i\mu_{\phi,i}}{M_\phi}\log\frac{\mu_i\mu_{\phi,i}}{M_\phi}.$$

Equation (2.11) can be applied immediately, obtaining

$$H(S) = \log M_\phi - \frac{1}{M_\phi}\sum_{i=1}^{n}\mu_i^\phi\mu_{\phi,i}\log\mu_i^\phi\mu_{\phi,i}. \tag{2.13}$$

The convention $0\log 0 = 0$ is applied here for disconnected words (where $\mu_i = 0$).

The marginal meaning entropy is found symmetrically, the marginal probability (Equation (2.9)) is applied to the definition of $H(R)$ (Equation (**??**)). With Equation (2.11), we obtain

$$H(R) = \log M_\phi - \frac{1}{M_\phi}\sum_{j=1}^{m}\omega_j^\phi\omega_{\phi,j}\log\omega_j^\phi\omega_{\phi,j}. \tag{2.14}$$

also using the convention $0\log 0 = 0$ for disconnected meanings ($\omega_j = 0$).

Dynamic Equations

The full derivation of the dynamic equations can be found in [1]. Here only the final expressions of the dynamic equations are given.

Starting with compact expressions of the entropies (Equations (2.12), (2.13) and (2.14))

$$H(S,R) = \log M_\phi - \frac{\phi}{M_\phi} X(S,R) \tag{2.15}$$

$$H(S) = \log M_\phi - \frac{1}{M_\phi} X(S) \tag{2.16}$$

$$H(R) = \log M_\phi - \frac{1}{M_\phi} X(R) \tag{2.17}$$

with

$$X(S,R) = \sum_{(i,j)\in E} x(s_i, r_j) \tag{2.18}$$

$$X(S) = \sum_{i=1}^{n} x(s_i) \tag{2.19}$$

$$X(R) = \sum_{j=1}^{m} x(r_j) \tag{2.20}$$

$$x(s_i, r_J) = (\mu_i \omega_j)^\phi \log \mu_i \omega_j \tag{2.21}$$

$$x(s_i) = \mu_i^\phi \mu_{\phi,i} \log \mu_i^\phi \mu_{\phi,i} \tag{2.22}$$

$$x(r_j) = \omega_j^\phi \omega_{\phi,j} \log \omega_j^\phi \omega_{\phi,j}. \tag{2.23}$$

A prime mark is used to indicate a new value of a certain variable after a mutation has taken place. A variable without a prime mark indicates the value before the mutation took place. Suppose that $a_{i,j}$ mutates. Then

$$a'_{i,j} = 1 - a_{i,j} \tag{2.24}$$

$$\mu'_i = \mu_i + (-1)^{a_{i,j}} \tag{2.25}$$

$$\omega'_j = \omega_j + (-1)^{a_{i,j}} \tag{2.26}$$

We define the set of neighbors of any word s_i

$$\Gamma_S(i) = \{\, r_j \mid (s_i, r_j) \in E \,\}, \tag{2.27}$$

and similarly the set of neighbors of any meaning r_j

$$\Gamma_R(j) = \{\, s_i \mid (s_i, r_j) \in E \,\}. \tag{2.28}$$

Then, for any k such that $1 \le k \le n$, we have that

$$\mu'_{\phi,k} = \begin{cases} \mu_{\phi,k} - a_{ij}\omega_j^\phi + (1-a_{ij}){\omega'_j}^\phi & \text{if } k = i \\ \mu_{\phi,k} - \omega_j^\phi + {\omega'_j}^\phi & \text{if } k \in \Gamma_R(j) \text{ and } k \neq i \\ \mu_{\phi,k} & \text{otherwise.} \end{cases} \tag{2.29}$$

Likewise, for any l such that $1 \leq l \leq m$, we have that

$$\omega'_{\phi,l} = \begin{cases} \omega_{\phi,l} - a_{ij}\mu_i^{\phi} + (1-a_{ij})\mu_i'^{\phi} & \text{if } l = j \\ \omega_{\phi,l} - \mu_i^{\phi} + \mu_i'^{\phi} & \text{if } l \in \Gamma_S(i) \text{ and } l \neq j \ . \\ \omega_{\phi,l} & \text{otherwise.} \end{cases} \tag{2.30}$$

These variables can be applied to Equations (2.21), (2.22) and (2.23) directly to obtain their values.

We define the set $N_{i,j}$ (originalment $E(i,j)$ però ja es fa servir E com el conjunt d'arestes del graf) as the set of all edges connectings_i and r_j with their neighbors. That is,

$$N_{i,j} = \{\, (i,l) \mid l \in \Gamma_S(i) \,\} \cup \{\, (k,j) \mid k \in \Gamma_R(j) \,\} \tag{2.31}$$

With these definitions, we can now obtain the expressions of $X'(S,R)$ from $X(S,R)$ and of M'_{ϕ} from M_{ϕ}.

$$\begin{aligned} M'_{\phi} = M_{\phi} - &\left[\sum_{(k,l)\in N_{i,j}} (\mu_k\omega_l)^{\phi}\right] - a_{ij}(\mu_i\omega_j)^{\phi} \\ &+ \left[\sum_{(k,l)\in N_{i,j}} (\mu'_k\omega'_l)^{\phi}\right] + (1-a_{ij})(\mu'_i\omega'_j)^{\phi}. \end{aligned}$$

Similarly, the new value of $X(S,R)$ will be

$$\begin{aligned} X'(S,R) = X(S,R) - &\left[\sum_{(k,l)\in N_{i,j}} x(s_k,r_l)\right] - a_{ij}x(s_i,r_j) \\ &+ \left[\sum_{(k,l)\in N_{i,j}} x'(s_k,r_l)\right] + (1-a_{ij})x'(s_i,r_j). \end{aligned} \tag{2.32}$$

The expressions of $X'(S)$ from $X(S)$ and $X'(R)$ from $X(R)$ used to dynamically update Equations (2.32), (2.33) and (2.34) are

$$X'(S) = X(S) - \left[\sum_{k\in\Gamma_R(j)} x(s_k)\right] - a_{ij}x(s_i) + \left[\sum_{k\in\Gamma_R(j)} x'(s_k)\right] + (1-a_{ij})x'(s_i) \tag{2.33}$$

and

$$X'(R) = X(R) - \left[\sum_{l\in\Gamma_S(i)} x(r_l)\right] - a_{ij}x(r_j) + \left[\sum_{l\in\Gamma_S(i)} x'(r_l)\right] + (1-a_{ij})x'(r_j). \tag{2.34}$$

Again, the full derivation of these equations can be found in [1] and it is not included here to avoid repeating the same ideas.

The values of $H(S)$, $H(R)$ and $H(S,R)$ can be obtained by applying Equations (2.32), (2.33) and (2.34) to the definitions in Equations (2.15), (2.16) and (2.17) respectively.

Extreme cases and invariants

For verification purposes (see Section 3.4), the values of the entropies along with their invariants are also given here.

These follow immediately from the formulas given in this section and can be easily verified.

Extreme cases

- For a single edge, $H(S), H(R), H(S,R) = 0$.
- For a complete graph, $H(S) = \log n, H(R) = \log m, H(S,R) = \log nm$
- For a one-to-one mapping of signals into meanings with $n = m$, $H(S), H(R), H(S,R) = \log n$

Invariants

- $0 \leq H(S) \leq \log n$
- $0 \leq H(R) \leq \log m$
- $0 \leq H(S,R) \leq \log nm$

2.1.3 The *a priori* model

This model was introduced in [3] with meaning probabilities being constant and disconnected meanings being disallowed. Here we present a generalization of the previous model. In this model, the probability of a meaning r_j depends on an *a priori* probability $\pi(r_j)$, instead of depending directly on the structure of the bipartite graph. In [3] disconnected meanings were disallowed and $\pi(r_j) = \frac{1}{m}$.

Here, disconnected meanings are allowed and $\pi(r_j)$ can be any distribution of probabilities. Zero values of $\pi(r_j)$ are disallowed, however, as meanings that would never occur are not the target of communication. As expected,

$$\sum_{j=1}^{m} \pi(r_j) = 1. \tag{2.35}$$

The probability of a meaning is defined in a way to ensure that

$$\sum_{j=1}^{m} p(r_j) = 1 \tag{2.36}$$

even when there are disconnected meanings whose probability should be zero,

$$p(r_j) = \frac{(1-\delta_{\omega_j,0})\pi(r_j)}{\rho} \tag{2.37}$$

where

$$\begin{aligned} \rho &= \sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j) \\ &= 1-\sum_{j=1}^{m}\delta_{\omega_j,0}\pi(r_j). \end{aligned} \tag{2.38}$$

It is immediate to see that Equation (2.36) holds.

Joint probability

referencia a introduccio on es citi [4], posar aquesta part abans del titol joint probability? The conditional probability of choosing a word given a meaning is proportional to the number of meanings associated with that word or zero if that meaning is not associated to the word,

$$p(s_i|r_j) \propto a_{i,j}\mu_i^{\phi}. \tag{2.39}$$

Applying

$$\sum_{i=1}^{n} p(s_i|r_j) = 1$$

to Equation (2.39) and recalling Equation (2.4) we obtain

$$p(s_i|r_j) = \frac{a_{i,j}\mu_i^{\phi}}{\omega_{\phi,j}}. \tag{2.40}$$

The joint probability $p(s_i, r_j)$ is obtained by applying Equation (2.37) and Equation (2.40) to the definition

$$p(s_i, r_j) = p(s_i|r_j)p(r_j),$$

obtaining

$$p(s_i, r_j) = \frac{a_{i,j}(1-\delta_{\omega_j,0})\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}}. \tag{2.41}$$

Marginal word probability

The probability of a word can be derived from Equation (2.41) and applying

$$p(s_i) = \sum_{j=1}^{m} p(s_i, r_j),$$

obtaining

$$p(s_i) = \frac{\mu_i^\phi \chi_i}{\rho} \tag{2.42}$$

with

$$\chi_i = \sum_{j=1}^{m} \frac{a_{i,j}(1-\delta_{\omega_j,0})\pi(r_j)}{\omega_{\phi,j}}. \tag{2.43}$$

Entropies

Applying the definition of $H(R)$ (Equation (**??**)) to $p(s_j)$ (Equation (2.37)) we obtain

$$H(R) = -\sum_{j=1}^{m} \frac{(1-\delta_{\omega_j,0})\pi(r_j)}{\rho} \log \frac{(1-\delta_{\omega_j,0})\pi(r_j)}{\rho}.$$

Equation (2.11) can be applied immediately to simplify $H(R)$

$$H(R) = \log \rho - \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j)\log(1-\delta_{\omega_j,0})\pi(r_j).$$

This is simplified further applying the convention $0 \log 0 = 0$

$$\begin{aligned} H(R) &= \log \rho - \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\log \pi(r_j) \qquad (2.44)\\ &= \log \rho + \frac{1}{\rho}\left(H_\pi(R) + \sum_{j=1}^{m}\delta_{w_j,0}\pi(r_j)\log \pi(r_j)\right) \end{aligned}$$

where $H_\pi(R)$ is the entropy of the *a priori* probabilities

$$H_\pi(R) = -\sum_{j=1}^{m} \pi(r_j)\log \pi(r_j)$$

$H(S,R)$ can be derived by applying Equation (2.41) to the information theory definition of $H(S,R)$ (Equation (**??**)). Alternatively, it can also be derived by using the equality

$$H(S,R) = H(S|R) + H(R).$$

This reduces the problem to finding $H(S|R)$ using

$$H(S|R) = \sum_{j=1}^{m} H(S|r_j)p(r_j)$$

and

$$H(S|r_j) = -\sum_{i=1}^{n} p(s_i|r_j)\log p(s_i|r_j).$$

Both approaches turn out to be quite cumbersome mathematically. For the sake of brevity they are not shown here and can be consulted in Appendix B.2. In either case, the resulting expression for $H(S,R)$ is

$$H(S,R) = \log\rho - \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\left[\frac{\phi\nu_j}{\omega_{\phi,j}} + \log\frac{\pi(r_j)}{\omega_{\phi,j}}\right] \tag{2.45}$$

with

$$\nu_j = \sum_{i=1}^{n} a_{i,j}\mu_i^{\phi}\log(\mu_i). \tag{2.46}$$

$H(S)$ is derived quite easily by applying Equation (2.42) to the information theory definition (Equation (**??**))

$$H(S) = -\sum_{i=1}^{n}\frac{\mu_i^{\phi}\chi_i}{\rho}\log\frac{\mu_i^{\phi}\chi_i}{\rho}$$

In Appendix B.3 it is shown that $\sum_{i=1}^{n}\mu_i^{\phi}\chi_i = \rho$, thus (2.11) can be applied here, obtaining

$$H(S) = \log\rho - \frac{1}{\rho}\sum_{i=1}^{n}\mu_i^{\phi}\chi_i\log\mu_i^{\phi}\chi_i \tag{2.47}$$

Dynamic Equations

The full derivation of the dynamic equations can be found in Appendix B.4. Here only the final expressions of the dynamic equations are given. Recall Section 2.1.2 for many useful definitions. In all dynamic equations, a mutation on $a_{i,j}$ is assumed.

Compact expressions of the entropies (Equations (2.44), (2.45) and (2.47))

$$H(S,R) = \log\rho - \frac{1}{\rho}X(S,R) \tag{2.48}$$

$$H(S) = \log\rho - \frac{1}{\rho}X(S) \tag{2.49}$$

$$H(R) = \log\rho - \frac{1}{\rho}X(R) \tag{2.50}$$

with

$$X(S,R) = \sum_{j=1}^{m}(1-\delta_{\omega_j,0})x(r_j) \tag{2.51}$$

$$X(S) = \sum_{i=1}^{n} x_{s_i} \tag{2.52}$$

$$X(R) = \sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j)\log \pi(r_j) \tag{2.53}$$

$$x(s_i) = \mu_i^{\phi}\chi_i \log\left(\mu_i^{\phi}\chi_i\right) \tag{2.54}$$

$$x(r_j) = \pi(r_j)\left[\frac{\phi\nu_j}{\omega_{\phi,j}} + \log\frac{\pi(r_j)}{\omega_{\phi,j}}\right]. \tag{2.55}$$

Equations (2.24) ($a'_{i,j}$) and (2.25) (μ'_i) remain the same as in the proportional model. Equation (2.26) (ω'_j) is unused in this model.

Equation (2.30) ($\omega'_{\phi,l}$ for any l such that $1 \le l \le m$) remains identical, while Equation (2.29) ($\mu'_{\phi,k}$ for any k such that $1 \le k \le n$) is unused in this model.

Recall as well the definition of the set of all edges to neighbors of s_i and r_j (Equation (2.31)). Two additional sets are defined for ease in the definition of these equations, $A_{i,j}(k)$ is the set of all meanings that are neighbors of i ($\Gamma_S(i)$) (plus the meaning r_j if not already included) that are also neighbors of k

$$A_{i,j}(k) = (\Gamma_S(i) \cup \{\, r_j \,\}) \cap \Gamma_S(k). \tag{2.56}$$

The other set, $B_{i,j}(k)$ is the set of all words s_k such that $A_{i,j}(k)$ is not the empty set. The word s_i is always included,

$$B_{i,j}(k) = \{\, s_k \mid A_{i,j}(k) \neq \emptyset \,\} \cup \{\, s_i \,\} \tag{2.57}$$

The one mutation change equations for ρ, ν and χ are

$$\rho' = \rho - \delta_{\omega'_j,0}\pi(r_j) + \delta_{\omega_j,0}\pi(r_j), \tag{2.58}$$

$$\nu'_l = \begin{cases} \nu_l + (1-a_{ij})\mu'^{\phi}_i \log\mu'_i - a_{ij}\mu_i^{\phi}\log\mu_i & \text{if } l = j \\ \nu_l - \mu_i^{\phi}\log\mu_i + \mu'^{\phi}_i\log\mu'_i & \text{if } l \in \Gamma_S(i) \text{ and } l \neq j \\ \nu_l & \text{otherwise} \end{cases} \tag{2.59}$$

and

$$\chi'_k = \chi_k - \sum_{l\in A_{i,j}(k)}\frac{\pi(r_l)}{\omega_{\phi,l}} + \sum_{l\in A_{i,j}(k)}\frac{\pi(r_l)}{\omega'_{\phi,l}}. \tag{2.60}$$

χ_i is always recalculated statically instead

These variables can be applied to Equations (2.54) and (2.55) directly to obtain their values.

The expressions of $X'(S)$, $X'(R)$ and $X'(S,R)$ used to dynamically update Equations (2.44), (2.45) and (2.47) are

$$X'(R) = X(R) - \delta_{\omega'_j,0}\pi(r_j)\log\pi(r_j) + \delta_{\omega_j,0}\pi(r_j)\log\pi(r_j), \tag{2.61}$$

$$X'(S,R) = X(S,R) - \sum_{l\in\Gamma_S(i)\setminus\{j\}} x(r_l) + \sum_{l\in\Gamma_S(i)\setminus\{j\}} x'(r_l) - (1-\delta_{\omega_j,0})x(r_j) + (1-\delta_{\omega'_j,0})x'(r_j) \tag{2.62}$$

and

$$X'(S) = X(S) - \sum_{o\in B_{i,j}(k)} x(r_o) + \sum_{o\in B_{i,j}(k)} x'(r_k) \tag{2.63}$$

Again, the full derivation and logic behind these equations can be found in Appendix B.4. It is not included here for brevity, as it is a long not immediately obvious derivation.

Extreme cases and invariants

For verification purposes (see Section 3.4), the values of the entropies along with their invariants are also given here.

These follow immediately from the formulas given in this section and can be easily verified.

Extreme cases

- For a single edge, $H(S), H(R), H(S,R) = 0$.
- For a complete graph, $H(S) = \log n, H(R) = H_\pi(R), H(S,R) = H_\pi(R) + \log n$
- For a one-to-one mapping of signals into meanings with $n = m$, $H(S), H(R), H(S,R) = H_\pi(R)$

Invariants

- $0 \le H(S) \le \log n$
- $0 \le H(R) \le H_\pi(R)$
- $0 \le H(S,R) \le H_\pi(R) + \log n$

2.2 Computational aspect

This section covers the computational side of the model. This side is based on the mathematics covered in Section 2.1. Both models are seen separately, Section 2.2.1 covers the proportional model while Section 2.2.2 covers the *a priori* model. Both sections cover the computational cost of computing the variables of the model, both completely (static calculation) and the change after a mutation to $a_{i,j}$ (dynamic calculation) as well as the changes that take place by treating the case $\phi = 0$ separately.

2.2.1 The proportional model

This section covers the computational cost of the calculation of the entropies of the proportional model.

Static

A, E, μ and ω are always calculated dynamically and updated whenever an edge is added or removed to the graph, as it is very simple to do so. All entropies are calculated statically in a single loop iterating over every edge in E. The joint entropy $H(S,R)$ (Equation (2.12)) and the normalization factor M_ϕ (Equation (2.6)) are updated on every step of the loop. μ_ϕ and ω_ϕ are updated on every iteration as well. $H(S)$ (Equation (2.13)) is only updated when $\mu_{\phi,i}$ for a particular word s_i has been fully recalculated. $H(R)$ (Equation (2.14)) is updated in the same way as $H(S)$, whenever $\omega_{\phi,j}$ for a particular meaning r_j has been fully recalculated. The cost of the static calculation of entropies is $\mathcal{O}(|E|)$.

Dynamic

In the case of the dynamic calculation of entropies, the algorithmic cost is dominated by the computation of $X'(S)$, $X'(R)$ and $X'(S,R)$ (Equations (2.32), (2.33) and (2.34)). It can be seen immediately that looping over all the neighbors of s_i and r_j is necessary in order to update all entropies. The cost of the dynamic calculation of entropies is then $\mathcal{O}(\max(|\Gamma_S(i)|, |\Gamma_R(j)|))$.

The case $\phi = 0$

In order to speed up calculations, simpler equations for the case $\phi = 0$ are derived. $X'(S,R)$ (Equation (2.32)) is no longer used, as $H(S,R) = \log M$ when $\phi = 0$ (see Equation (2.12)).

$X'(S)$ (Equation (2.33)) becomes

$$X'(S) = X(S) - \mu_i \log \mu_i + \mu'_i \log \mu'_i \tag{2.64}$$

using the convention $0 \log 0 = 0$ where necessary.

Similarly to $X'(S)$, $X'(R)$ (Equation (2.34)) becomes

$$X'(R) = X(R) - \omega_j \log \omega_j + \omega'_j \log \omega'_j \tag{2.65}$$

In the case $\phi = 0$, M_ϕ (Equation (2.6)) becomes $|E|$, the number of edges in the graph.

As can be seen from equations (2.64) and (2.65), the cost of the dynamic calculation is greatly reduced when $\phi = 0$, becoming $\mathcal{O}(1)$.

2.2.2 The *a priori* model

This section overs the computational cost of the calculation of the entropies of the *a priori* model.

Static

As with the proportional model (see Section 2.2.1), A, E, μ and ω are dynamically updated whenever an edge is added or removed from the graph while all other variables, including entropies, are calculated statically.

As can be seen from Equations (2.44), (2.45) and (2.47), a loop over every edge in the graph is needed in order to recalculate all entropies and variables. Unlike the proportional model, however, this loop needs to be repeated twice.

During the first run, ω_ϕ (Equation (2.4)) and ν (Equation (2.46)) are updated on every iteration. ρ (Equation (2.38)), $H(R)$ (Equation (2.44)) and $H(S,R)$ (Equation (2.45)) are calculated only on the iterations where $\omega_{\phi,j}$ and ν_j have been completely calculated for a single meaning r_j.

This leaves χ and $H(S)$ to be calculated during the second run of the loop over all edges. The computation of χ_i (Equation (2.43)) for a word s_i requires $\omega_{\phi,j}$ to be fully computed for every meaning r_j. This is the reason for running two separate loops, fully calculating ω_ϕ in one run so that χ may be calculated in the other. $H(S)$ (Equation (2.47)) is updated only on the iterations where χ_i has been completely calculated for a single word s_i.

The cost of the static calculation of entropies is the same as in the proportional model, $\mathcal{O}(|E|)$.

Dynamic

The dynamic calculation of entropies is dominated by the computation of $X'(S)$, $X'(S,R)$ and χ (Equations (2.51), (2.52) and (2.60)). These three values iterate on three different sets: $\Gamma_S(i)$, $A_{i,j}$ and $B_{i,j}$ (Equations (2.27), (2.56) and (2.57)). It is clear from the definition of set $A_{i,j}$ is lesser or similar (one more element) in size to the set $\Gamma_S(i)$, but $\Gamma_S(i)$ will usually be bigger. The size of the set $B_{i,j}$ depends on the structure of the graph.

The cost of the dynamic calculation of entropies is then $\mathcal{O}(\max(|\Gamma_S(i)|, B_{i,j}))$

The case $\phi = 0$

In order to speed up calculations, simpler equations for the case $\phi = 0$ are derived. Much of the dependency on neighboring nodes is removed.

$X(R)$ and ρ (Equations (2.61) and (2.58)) are not affected as they do not depend on ϕ. They remain simple.

$x(r_j)$ becomes

$$(1-\delta_{w_j,0})\pi(r_j)\log\frac{\pi(r_j)}{\omega_j} \tag{2.66}$$

and so no longer depends on neighbors of r_j. Consequently, $X(S,R)$ also no longer depends on a set of neighbors and becomes

$$X'(S,R) = X(S,R) - x(r_j) + (1-\delta_{\omega'_j,0})x'(r_j). \tag{2.67}$$

χ_i (Equation (2.43)) is simplified but still depends on the neighborhood of s_i in order to be updated

$$\chi'_k = \begin{cases} \chi_i + (1 - a_{i,j}) \frac{\pi(r_j)}{\omega'_j} - a_{i,j} \frac{\pi(r_j)}{\omega_j} & \text{if } k = 1 \\ \chi_k - \frac{\pi(r_j)}{\omega_j} + \frac{\pi(r_j)}{\omega'_j} & \text{if } k \in \Gamma_R(j) \text{ and } k \neq i \\ \chi_k & \text{otherwise.} \end{cases} \tag{2.68}$$

As $X(S)$ depends on χ (Equation (2.52)), $X'(S)$ becomes

$$X'(S) = X(S) - \sum_{k \in \Gamma_R(j) \cup \{ i \}} x(s_i) + \sum_{k \in \Gamma_R(j) \cup \{ i \}} x'(s_i) \tag{2.69}$$

with (see Equation (2.54))

$$x(s_i) = \chi_i \log \chi_i. \tag{2.70}$$

Chapter 3

Methods

This chapter centers on the implementation of the models defined in Chapter 2 and the optimization method outlined in Section **??**. Section 3.1 focuses on the implementation of the models in a C++ program, including a class hierarchy diagram, several important algorithms, various methods used to avoid floating point error and the different distributions of π (see Section 2.1.3) that were implemented. Section 3.2 deals with the optimization algorithm, how it changes for the static or dynamic cases and several optimizations that allow for faster runtime and more precise results. On Section 3.3 the approach taken to parallelization is outlined and justified. Section 3.4 deals with the verification of the code. Finally, section 3.5 explains the main challenges involved with the implementation.

3.1 Model implementation

This section covers the implementation of the two models seen in Chapter 2. It starts with a class hierarchy diagram of the organization of the static and dynamic versions of the two models. Following this, the algorithm used to randomly generate a random bipartite graph (given a vertex probability or given the number of edges) is outlined. Following this, the various "tricks" used in order to minimize the floating point error, specially in the dynamic version of the model are explained. To close this section, details on the various distributions of π that were implemented are given.

Class hierarchy diagram

The diagram in Figure 3.1 shows the classes implementing the bipartite graph models (see Chapter 2).

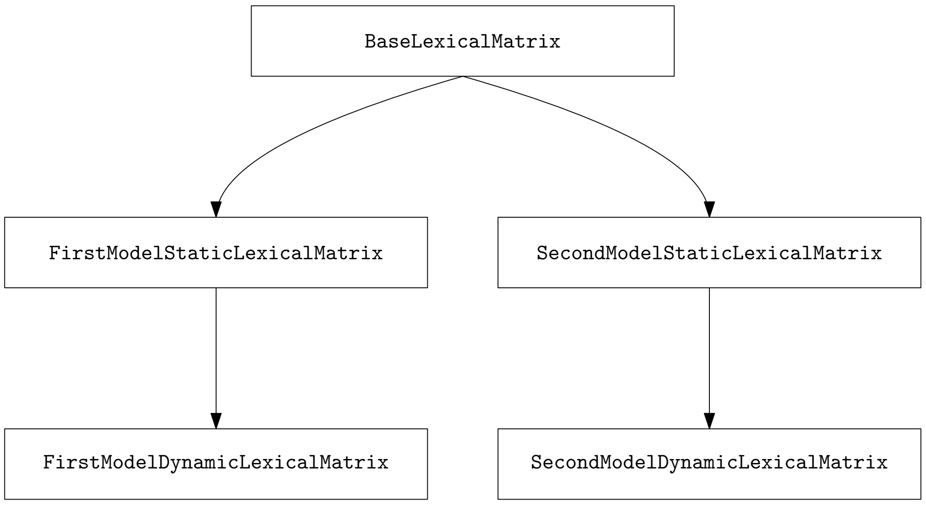

BaseLexicalMatrix
FirstModelStaticLexicalMatrix
SecondModelStaticLexicalMatrix
FirstModelDynamicLexicalMatrix
SecondModelDynamicLexicalMatrix

Figure 3.1: Diagram showing the hierarchy of the classes implementing the two models in their dynamic and static variants in the C++ program.

Random graph generation

This section outlines the algorithm implemented to generate a random Erdos-Renyi bipartite graph in pseudocode. The graph can be generated by specifying either the probability of edge creation ($G_{n,m,p}$) as seen in Algorithm 2 or the total number of edges ($G_{n,m,e}$) as in Algorithm **??**. Note that all Algorithm 2 actually does is compute a value of e then call Algorithm **??**.

citar alguna font? explicar d'on surt la notació $G_{n,m,e}$

Algorithm 1 $G_{n,m,p}$ algorithm to generate a random $n \times m$ graph given probability of edge p.

```
1: procedure $G(n, m, p)$
2:     $e \leftarrow \text{BINOM}(n \cdot m,\ p)$
3:     return $\text{G}(n, m, e)$
4: end procedure
```

Dealing with floating point error

One of the major challenges (see Section 3.5) has been dealing with floating point error. Two major "tricks" have been used in order to minimize this loss of precision as much as possible.

The Accumulator class Whenever possible, specially in the static calculations, additions have been done through an accumulator class. This class main-

Algorithm 2 $G_{n,m,e}$ algorithm to generate a random $n \times m$ graph given the number of edges e.

```
1: procedure $G(n, m, e)$
2:     if disconnected meanings disallowed then
3:         for each $r_j$ such that $1 \leq j \leq m$ do
4:             $s_i \leftarrow$ UNIFORM($s_1$, $s_n$)
5:             ADD EDGE($s_i, r_j$)
6:             $e \leftarrow e - 1$
7:         end for
8:     end if
9:     while $e > 0$ do
10:        $s_i \leftarrow$ UNIFORM($s_1$, $s_n$)
11:        $r_j \leftarrow$ UNIFORM($r_1$, $r_n$)
12:        ADD EDGE($s_i, r_j$)
13:        $e \leftarrow e - 1$
14:    end while
15: end procedure
```

tains two variables, `total` and `current`. When a value is added, it is added to the `current` variable. If the `current` variable is greater than `total`, it is added to `total` and reset to 0. This is an inexpensive and effective way to minimize the loss of precision from a common occurrence during static calculation, the addition of a smaller value into a greater one.

Static recalculation During the optimization process, whenever a new minimum is reached, a static recalculation is forced. In this way, one of the major source of floating point error from the dynamic version is removed. The dynamic equations (see Section 2.1) consist of many additions and subtractions. These additions and subtractions are a major contributor to the accumulation of floating point error. Since finding a new minimum is not common, this does not affect the run time of the optimization process.

Save and restore Most steps of the optimization process will not improve the cost function (see Section 3.2). Instead of undoing the step and recalculating all the parameters of the model, the parameters are saved before performing the step, and they are restored when the step needs to be reverted. This strategy reduces the amount of additions and subtractions to be done when the recalculation of parameters is dynamic, thus reducing the amount of floating point error.

Distribution of π

The second model (see Section 2.1.3) is based on a vector of length m of *a priori* probabilities π. Uniform, geometric, power law and broken stick probability distributions have been implemented in order to give values to the elements of this vector.

Uniform distribution The uniform distribution simply assigns the probability $1/m$ to every element in the vector.

Geometric distribution Assigns a right truncated geometric distribution with parameter p to the elements of π, following

$$(1-p)^{x-1}\frac{p}{1-(1-p)^m}.$$

This formula can be easily derived from the more general formula for a truncated discrete probability distribution

$$p(X=x)/(P(X=b)-P(X=a)).$$

Power law distribution Assigns a right truncated power law distribution with parameter α to the elements of π. It follows

$$x^{-\alpha}\sum_{i=1}^{m} i^{-\alpha}$$

Broken stick distribution A broken stick distribution corresponds to the distribution of the sizes of the pieces of a stick that is broken by a point along its length randomly a number $m-1$ of times. This distribution corresponds to the expected value of the lengths of the pieces of a stick of total length 1. The formula and its derivation can be found in [5] (Equation 1).

3.2 Optimization

The optimization algorithm follows the Markov chain Monte Carlo method at zero temperature. The model is initialized in some way (depending on parameters), and on each iteration a number of random mutations (adding or removing an edge on the underlying bipartite graph) are effectuated. The cost function is evaluated with a parameter λ. If it is found to have improved, the model is recalculated statically (to avoid floating point error, see Section 3.1). If it is found to not have improved, the changes are reverted. The optimization stops after a number of failures to improve the cost function. The optimization stops early if the global minimum for the cost function is reached.

The parameter λ is given as a parameter. It controls the weight of the two forces that contribute to the cost function. See Section **??**.

The number of mutations done on each step depends on the parameters and may be either constant or follow a binomial distribution. In the case of a binomial distribution, it is possible that zero mutations may be effectuated. In this case, nothing is done and the step is considered to not have improved the cost function.

The number of failures to improve the cost function needed to stop iterating depends on parameters. Two possible values are calculated from parameters of the model:

- **weak**: Enough steps are performed such that it is likely that all edges have been visited once. This follows the Coupon Collector's Problem, and the formula for the number of attempts is

$$\lfloor nm \log nm \rfloor$$

- **strong**: Enough steps are performed such that it is likely that every possible pair of edges has been visited once. This also follows the Coupon Collector's Problem,

$$\left\lfloor \binom{nm}{2} \log \binom{nm}{2} \right\rfloor$$

3.3 Parallelization

During the initial phases of the design of the software, it was considered to add parallelization in order to speed up calculations. Parallelization would add overhead and complexity to an already complex codebase in exchange for sharing the computation between the machine's processors.

A typical task for the program is the computation of a range of values of λ (see Section 3.2) and to calculate several samples for each λ. This layout lends itself well to multiprocessing, without the program implementing any sort of parallelization scheme.

The user is expected to divide the load his or herself, for instance by generating many separate configuration files (or using a script to do so) and using a tool such as one of the many implementations of `parallel` or a computing cluster's workload manager. In this way, the operating system handles the parallelization complexity and the software's complexity and overhead is reduced. This also makes verification much easier.

3.4 Verification

Verification is a vital part of any software development process. In this case, verification involves ensuring that both static and dynamic versions of the model are correctly implemented. Dynamic versions are specially complex and so need to be tested thoroughly.

The test routines implemented verify that the models give "sane" results by checking extreme cases (see Sections). In addition, the invariants of the entropies (also in Sections) are verified after every mutation and the program is aborted if they ever do not hold. A test also checks the invariants exhaustively for every possible combination of connected and disconnected edges for very small graphs ($n = m = 4$).

Other functionality less related with the implementation of the model is also tested.

- The save/restore functionality described in Section 3.1 for dealing with floating point error.
- Quickly testing whether the matrix has disconnected meanings (configuration parameter disallows disconnected meanings).
-

The last test is running the program for relatively small values of n and m and all combinations of parameters, and exhaustively checking that both static and dynamic versions give identical output for the same seed. This test is used to ensure the correctness of the dynamic version.

3.5 Challenges

Challenges and problems related to the implementation of the model are given in this section.

In both the static and dynamic versions of the calculations, there are many additions or subtractions of floating point numbers. These actions often involve a loss of precision due to the difference in magnitude of the values being operated. Over many iterations, this loss of precision can become apparent and change the final result of the simulation. See Section 3.1 where an entire subsection is dedicated to dealing with numerical error.

Since the runtime of the dynamic versions often depends on the number of neighbors of the affected vertices (see Section 2.2), running calculations on complete or almost complete graphs takes a long time, as the advantages of the dynamic formulas are reduced significantly. As a result of this, running the optimization with an initial complete graph can take an unacceptable amount of time when compared to other initial conditions.

The strong stop condition (Section 3.2) results far too many attempts to improve the cost function when $n = m$ is large (for $n = m = 400$, the number of attempts is of the order of 10^{12}). For this reason, the strong stop condition is not used in Chapter **??**

The geometric distribution decreases in probability very quickly. Because of this, erroneous values can be produced when the value of π is too close to 0. As outlined in Section 2.1.3, the values of π should never be zero. This problem occurs in general whenever π is too close to zero and the program automatically warns about it if the parameters will generate values of π that are too small. For this reason, the geometric distribution does not appear in Chapter **??**

Chapter 4

Results

4.1 verify results from old papers

- No hi ha massa info sobre parametres en articles Ramon

4.1.1 2005, first model

4.1.2 2003, second model

4.2 new results with bigger n=m

4.2.1 first model

- $\phi \in \{0, 1\}$
- $n = m = 400$
- $init \in \{\text{random}, \text{single}, \text{one-to-one}\}$
  - no complete
- stop condition = weak
- mutations = 2
- allow disconnected meanings

4.2.2 second model

- $\phi \in \{0, 1\}$
- $n = m = 400$
- $init \in \{\text{random}, \text{single}, \text{one-to-one}\}$

  - no complete
- $\pi \in \{\text{uniform}, \text{broken stick}, \text{power law}\}$
  - no geometric
- stop condition = weak
- mutations = 2
- allow disconnected meanings

Chapter 5

Discussion

Aquesta part és important, ha de quedar clara. Parlar tambe sobre limitacions: quins resultats son contundents i quins febles.

Analisi critic, interpretar, lligar amb context de coneixements: relacio paraula edat frequencia i altres models de zipf com es relacion amb ells, com ho expliquen. Distincio entre model que explica i model que descriu. No hi ha explicació sense teoria. Teoria es una serie de principis. No tots els models son explicacions pk no poden fer prediccions. E.g., random typing no diu res de significats no pot explicar aprenentatge en nens.

5.1 mates (linguistica quantitativa)

5.1.1 zipf law en regressio lineal del log-log?

- mirar grafiques dels dos models, variants

5.1.2 relacio edat-frequencia

- grafiques mostren que paraules mes antigues son mes frequents
- correspon amb zipf (llibre)

5.2 computacional

- lligar amb biosemiotics

5.2.1 minim local

- el llei zipf minim local? stop condition
- fins a quin punt recuperar lleis depen de parametres?

5.3 future work

5.3.1 altres metodes d'optimitzacio

- simulated annealing (non zero temperature)
- gradient descent (make Ω derivable)

5.3.2 altres valors de phi

hem fet 1 i 0 per altres, 0.5, 1.5...

5.3.3 vocabulary nens segon model

biosemiotics ha fet primer model nomes

5.3.4 evitar error numeric encara mes

- subcalculs del metode dinamic que es podrien fer estatics
- ja es fa un valor de χ

Appendix A

Codi

- afegir repositori amb el codi

Appendix B

Formulae

derivacions llargues i pagines i pagines de mates van aqui

B.1 Simplification of the equations of entropies

Equation (2.11) is used in several parts of the thesis to simplify the expressions of entropies. The derivation is short, but it is not added into the main text of the thesis for the sake of clarity and focus.

$$
\begin{gathered}
-\sum_i \frac{x_i}{T} \log \frac{x_i}{T} \\
-\frac{1}{T} \sum_i x_i \log x_i - x_i \log T \\
-\frac{1}{T} \sum_i x_i \log x_i + \log(T) \frac{1}{T} \sum_i x_i
\end{gathered}
$$

at this point we can apply

$$\sum_i x_i = T$$

and obtain

$$\log T - \frac{1}{T} \sum_i x_i \log x_i$$

B.2 Derivation of the joint entropy in the *a priori* model

In Section 2.1.3 the derivation of the joint entropy is left as just the high level reasoning of which formulas may be used to obtain its expression. Here, the full derivation of both approaches is given.

The first approach consists of applying Equation (2.41) to the information theory definition of $H(S,R)$ (Equation (**??**)),

$$
\begin{aligned}
H(S,R) &= -\sum_{i=1}^{n}\sum_{j=1}^{m} p(s_i,r_j)\log\left[p(s_i,r_j)\right] \\
&= -\sum_{i=1}^{n}\sum_{j=1}^{m} \frac{a_{i,j}(1-\delta_{\omega_j,0})\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}} \log\left[\frac{a_{i,j}(1-\delta_{\omega_j,0})\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}}\right] \\
&= -\frac{1}{\rho}\sum_{i=1}^{n}\sum_{j=1}^{m} \frac{a_{i,j}(1-\delta_{\omega_j,0})\mu_i^{\phi}\pi(r_j)}{\omega_{\phi,j}} \log\left[\frac{\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}}\right] \\
&= -\frac{1}{\rho}\sum_{j=1}^{m} \frac{(1-\delta_{\omega_j,0})\pi(r_j)}{\omega_{\phi,j}} \left[\phi\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi}\log\mu_i\right. \\
&\quad + \log\pi(r_j)\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi} - \log(\rho)\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi} \\
&\quad \left. - \log(\omega_{\phi,j})\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi}\right].
\end{aligned}
$$

Applying Equation (2.46) we arrive at Equation (2.45).

The second approach consists on using other less complex expressions we have available and building up from them using the definitions

$$H(S,R) = H(S|R) + H(R), \tag{B.1}$$

$$H(S|R) = \sum_{j=1}^{m} H(S|r_j)p(r_j), \tag{B.2}$$

$$H(S|r_j) = -\sum_{i=1}^{n} p(s_i|r_j)\log p(s_i|r_j). \tag{B.3}$$

Starting from Equation (B.3) and applying Equation (2.40) we obtain

$$
\begin{aligned}
H(S|r_j) &= -\sum_{i=1}^{n} \frac{a_{i,j}\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}} \log\left(\frac{a_{i,j}\mu_i^{\phi}\pi(r_j)}{\rho\omega_{\phi,j}}\right) \\
&= \log\omega_{\phi,j} - \frac{1}{\omega_{\phi,j}}\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi}\log(a_{i,j}\mu_i^{\phi}) \\
&= \log\omega_{\phi,j} - \frac{\phi}{\omega_{\phi,j}}\sum_{i=1}^{n} a_{i,j}\mu_i^{\phi}\log(\mu_i)
\end{aligned}
$$

and after applying Equation (2.46)

$$
H(S|r_j) = \log\omega_{\phi,j} - \frac{\phi\nu_j}{\omega_{\phi,j}}. \tag{B.4}
$$

Applying Equations (B.4) and (2.37) to Equation (B.2) we reach

$$
\begin{aligned}
H(S|R) =& \sum_{j=1}^{m}\left(\log\omega_{\phi,j} - \frac{\phi\nu_j}{\omega_{\phi,j}}\right)\frac{(1-\delta_{w_j,0})\pi(r_j)}{\rho} \\
=& \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\left[\log(\omega_{\phi,j}) - \frac{\phi\nu_j}{\omega_{\phi,j}}\right].
\end{aligned} \tag{B.5}
$$

The last step consists of applying Equations (B.5) and (2.44) to Equation (B.1), obtaining Equation (2.45) again

$$
\begin{aligned}
H(S,R) = & \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\left[\log(\omega_{\phi,j}) - \frac{\phi\nu_j}{\omega_{\phi,j}}\right] \\
& + \log\rho - \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\log\pi(r_j) \\
& = \log\rho - \frac{1}{\rho}\sum_{j=1}^{m}(1-\delta_{w_j,0})\pi(r_j)\left[\frac{\phi\nu_j}{\omega_{\phi,j}} + \log\frac{\pi(r_j)}{\omega_{\phi,j}}\right].
\end{aligned}
$$

B.3 Proof that $\sum_{i=1}^{n}\mu_i^{\phi}\chi_i = \rho$

In order to use Equation (2.11) in order to simplify the derivation of Equation (2.47), it must hold that $\sum_{i=1}^{n}\mu_i^{\phi}\chi_i = \rho$. Here it is shown that this is the case.

Starting from the equality, we can expand both sides

$$\begin{aligned}
\sum_{i=1}^{n} \mu_i^{\phi} \chi_i &= \rho \\
\sum_{i=1}^{n} \mu_i^{\phi} \sum_{j=1}^{m} \frac{a_{ij}(1-\delta_{\omega_j,0})\pi(r_j)}{\omega_{\phi,j}} &= \sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j) \\
\sum_{j=1}^{m} \left(\sum_{i=1}^{n} a_{ij}\mu_i^{\phi} \right) \frac{(1-\delta_{\omega_j,0})\pi(r_j)}{\omega_{\phi,j}} &= \sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j),
\end{aligned}$$

applying Equation (2.4) we obtain

$$\sum_{j=1}^{m}(1-\delta_{\omega_j,0})\pi(r_j)$$

on both sides.

Recall the definition of $p(s_i)$ in Equation (2.42) to see that

$$\sum_{i=1}^{n} p(s_i) = 1$$

immediately follows.

B.4 Dynamic Equations for the *a priori* model

The full derivation of the dynamic equations for the *a priori* model is given in this section. The equations themselves as well as other definitions used here are already given in Section 2.1.3.

In this section, the derivation is logical rather than mathematical, consisting of explanations behind the logic of the equations.

ρ (see Equation (2.38)) only changes when, as a result of the mutation, the meaning r_j either was connected and becomes disconnected or was disconnected and becomes connected, resulting in Equation (2.58).

ν_l is very similar to $\omega_{\phi,l}$ (compare Equations (2.46) and (2.4)). Equation (2.59) shows how ν_l changes in the same cases and in the same way as $\omega_{\phi,l}$ (Equation (2.30)).

χ_k depends on $\omega_{\phi,l}$ such that $l \in \Gamma_S(k)$ (see Equation (2.43)). For χ_i, the entire value has to be recalculated, as every $\omega_{\phi,l}$ for $l \in \Gamma_S(k)$ will have changed (Equation (2.30)). It is more efficient (and reduces the amount of floating point error) to calculate χ_i statically than to subtract every $\omega_{\phi,l}$ and then add every $\omega'_{\phi,l}$. For all other values of χ_k ($k \neq i$), the affected $\omega_{\phi,l}$ are the intersection of the set of neighbors of i plus the meaning r_j and the set of neighbors of k. This is the set $A_{i,j}(k)$ defined in Equation (2.56). With this reasoning we reach the formula for the dynamic recalculation of χ_k, Equation (2.60).

$X(R)$ (Equation (2.53)) changes in a very similar way to ρ (Equation (2.38)), as they both depend on the same variable in the same way: only when the meaning r_j becomes connected or disconnected does $X(R)$ change, resulting in Equation (2.58).

$X(S,R)$ (Equation (2.51)) will change when a $x(r_l)$ (Equation (2.55)) changes. Both $\omega_{\phi,l}$ and ν_l change for $l \in \Gamma_S(i) \cup \{j\}$ (Equations (2.30) and (2.59)). So these are the components $x(s_l)$ of $H(S,R)$ that will change. For the special case of $l = j$ we should not subtract the old value when r_j has become connected as a result of the mutation (as that component was not present), and we should not add the new value when r_j has become disconnected (as that component should not be present). Or in other words, only subtract the old value if r_j was connected before the mutation (this serves to either update it or to remove it) and only add the new value if r_j is connected after the mutation (this serves to either update it or to add it). The resulting expression for $X'(S,R)$ is in Equation (2.62).

$X(S)$ (Equation (2.52)) will change when $x(s_k)$ (Equation (2.54)) changes, which depends on χ_k (Equation (2.60)). χ_k is updated for every k such that $|A_k| \neq \emptyset$ and χ_i is always updated, which is the definition of the $B_{i,j}(k)$ set (Equation (2.57)). Using $B_{i,j}(k)$ we obtain the expression for $X'(S)$ in Equation (2.63).

Bibliography

[1] David Carrera-Casado and Ramon Ferrer-i-Cancho. “The advent and fall of a vocabulary learning bias from communicative efficiency”. In: *Biosemiotics* (2021). URL: `http://arxiv.org/abs/2105.11519`.

[2] Ramon Ferrer-i-Cancho. “Zipf’s law from a communicative phase transition”. In: *European Physical Journal B* 47.3 (2005), pp. 449–457. DOI: `10.1140/epjb/e2005-00340-y`.

[3] Ramon Ferrer-i-Cancho and Ricard V. Sole. “Least effort and the origins of scaling in human language”. In: *Proceedings of the National Academy of Sciences of the United States of America* 100.3 (2003), pp. 788–791. DOI: `10.1073/pnas.0335980100`.

[4] Ramon Ferrer-i-Cancho and Michael S. Vitevitch. “The origins of Zipf’s meaning-frequency law”. In: *Journal of the Association for Information Science and Technology* 69.11 (Nov. 2018), pp. 1369–1379. DOI: `10.1002/asi.24057`.

[5] J. S. Smart. “Statistical tests of the broken-stick model of species-abundance relations”. In: *Journal of Theoretical Biology* 59.1 (1976), pp. 127–139. DOI: `10.1016/S0022-5193(76)80027-6`.