UNIVERSITAT DE BARCELONA
Facultat de Matemàtiques i Informàtica

GRAU D'ENGINYERIA INFORMÀTICA I MATEMÀTIQUES

Treball final de grau

Models estadístics i neuronals per a traducció automàtica : n-grames, models ocults de Màrkov i Transformadors

Autor: Manel Queral Martínez

Director:	**Dra. Maria Carmen Florit Selma**
	Dr. Santiago Seguí Mesquida
Realitzat a:	**Departament de Matemàtiques i Informàtica**

Barcelona, **14 de gener de 2025**

Abstract

Machine translation is the process of generating a translated output text in a target language from an initial input text in a source language without human intervention. Early machine translation systems were based on statistical models that calculated the probabilities of an output sentence given the initial input sentence, selecting as the translation the one with the highest probability.

The origins of this field lie in the area of Information Theory, where techniques designed to model message transmission through channels were adapted to capture the nature of language and provide a technical approach to tasks that had previously been considered purely humanistic.

The first models of this type were based on sequences of n-words, known as n-grams, which eventually evolved into models based on Markov chains that analyzed transitions between elements. These later models allowed the implicit inference of syntactic and grammatical structures.

With the advent of artificial intelligence techniques and the general improvement of hardware, statistical models were replaced by neural models based on neural networks, which promised better results through learning methods and massive training datasets.

In 2017, the introduction of the Transformer model marked a significant qualitative leap in the efficiency of these systems, forming the foundation of today's state-of-the-art models, which produce remarkable results.

The objective of this work is to provide an introductory analysis of various statistical models used in the early days of machine translation as well as the Transformer model, offering a comparative view of the evolution in the techniques employed. To support this analysis, an introduction to probability concepts and information theory, which are crucial for understanding the functioning of the studied models, is also provided.

Additionally, the implementation of a variant of the Transformer, MarianMT, is presented as an instructive example of a modern training pipeline.

Resum

La traducció automàtica és el procés de generar un text de sortida en un idioma destí a partir d'un text inicial d'entrada en un idioma origen sense intervenció humana. Els primers programes de traducció automàtica estaven basats en models estadístics que calculaven les probabilitats d'una frase d' *output* a partir de la frase donada inicialment, donant com a traducció aquella amb probabilitat màxima.

L'origen d'aquest camp prové de l'àrea de la Teoria de la Informació, on les tècniques aplicades per a modelar la transmissió de missatges a través de canals van ser adaptades per a poder capturar la naturalesa del llenguatge i oferir una aproximació tècnica a tasques que fins al moment s'havien considerat purament humanístiques.

Els primers models d'aquest tipus van ser basats en seqüències de n-paraules anomenades n-grames, que a la llarga van evolucionar en models basats en cadenes de Màrkov segons la transició entre elements. Aquests últims van permetre començar a inferir de manera implícita estructures sintàctiques i gramaticals.

Amb l'adveniment de les tècniques d'intel·ligència artificial i la millora general del

hardware, els models estadístics van ser substituïts per models neuronals, basades en xarxes neuronals que prometien obtenir millors resultats a partir de mètodes d'aprenentatge i conjunts de dades d'entrenament massius.

El 2017 la introducció del model Transformador (*Transformer*) es va generar un salt qualitatiu notable en l'eficiència d'aquests models, generant la base dels models que actualment es consideren l'*state-of-the-art* que produeixen resultats sorprenents.

L'objectiu d'aquest treball és fer una anàlisi introductòria diversos models estadístics emprats en els inicis de la traducció automàtica i també del model Transformador; per a poder oferir una visió comparativa de l'evolució en les tècniques emprades. Com a base en això, s'ofereix una introducció a conceptes de probabilitats i teoria de la informació que són crucials per a entendre el funcionament dels models estudiats.

A més, s'ofereix una implementació d'una variant del transformador *MarianMT* com a exemple instructiu d'un *pipeline* d'entrenament modern.

2020 Mathematics Subject Classification. 68T50, 94A15, 60J20

Agraïments

Vull agrair als meus tutors, Santi i Carmen, per haver-me guiat durant el procés.

Índex

1 Introducció

La traducció automàtica és una de les aplicacions més fascinants dels mètodes computacionals al processament del llenguatge. Té com a objectiu generar un text traduït en un idioma destí a partir d'un text d'entrada en un idioma origen, sense necessitat d'intervenció humana. Els orígens d'aquest camp es remunten a mitjans del segle XX, quan els primers sistemes es van inspirar en conceptes de la Teoria de la Informació. Aquests sistemes buscaven modelar el llenguatge com una estructura matemàtica, adaptant tècniques inicialment dissenyades per a la transmissió de missatges a través de canals sorollosos.

La primera generació de models de traducció automàtica tenia un caràcter estadístic i es centrava a calcular les probabilitats de les frases de sortida a partir de les frases d'entrada. Mitjançant el modelatge de la traducció com un procés probabilístic, aquests sistemes podien identificar de manera sistemàtica la traducció més probable. Una idea fonamental en aquest enfocament era l'ús dels n-grames, seqüències de n paraules, que permetien als models capturar patrons i dependències en el llenguatge. Aquesta metodologia va evolucionar amb la introducció de les cadenes de Màrkov, que van permetre modelar les transicions entre elements lingüístics, possibilitant la inferència implícita d'estructures gramaticals i sintàctiques.

L'arribada de les tècniques d'intel·ligència artificial i de les xarxes neuronals a principis del segle XXI va revolucionar la traducció automàtica, substituint els mètodes estadístics per models neuronals. Aquests nous sistemes van aprofitar conjunts massius de dades d'entrenament i algoritmes avançats d'aprenentatge per assolir nivells sense precedents d'exactitud i fluïdesa en les tasques de traducció. En particular, la introducció del model Transformador (*Transformer*) l'any 2017 va suposar un punt d'inflexió, establint un nou estàndard d'eficiència i rendiment en aquest camp. Aquesta arquitectura, caracteritzada pels seus mecanismes d'atenció i la seva capacitat de processament en paral·lel, constitueix la base dels sistemes *state-of-the-art* actuals.

Aquest treball explora l'evolució de la traducció automàtica des dels models estadístics fins als models neuronals. La primera part introdueix els conceptes matemàtics fonamentals, com les probabilitats i la teoria de la informació, que formen la base dels mètodes estadístics inicials. A continuació s'analitza l'estructura i el funcionament del model Transformador ,a més, per oferir una perspectiva pràctica, s'inclou la implementació d'una variant del Transformador, MarianMT, com a exemple instructiu d'un *pipeline* d'entrenament modern.

Mitjançant l'examen dels aspectes teòrics i pràctics de la traducció automàtica, aquest treball busca proporcionar una comprensió comparativa de la progressió des dels enfocaments probabilístics fins a les tècniques neuronals contemporànies. L'objectiu final és permetre al lector adquirir els coneixements bàsics del funcionament dels sistemes de traducció automàtica d'ús habitual avui en dia a més d'oferir una perspectiva històrica de l'evolució de la traducció automàtica.

Estructura de la Memòria

La memòria consta de cinc capítols i un apèndix (sense comptar la introducció). Els dos primers capítols serveixen per introduir els conceptes de probabilitats i teoria de la informació, respectivament ; que resulten útils per entendre la resta del treball.

Els dos següents introdueixen els models estadístics i neuronals per a la traducció au-

tomàtica. Els estadístics es basen en la modelització per n-grames i cadenes de Màrkov, mentre que els neuronals ofereixen una breu explicació dels models seqüencials i el Transformador, basat en atenció.

Finalment s'ofereix una breu conclusió sobre el treball realitzat.

Addicionalment, s'afegeix un apèndix donant detalls tècnics l'estructura del model Transformador i la implementació realitzada del *MarianMT* per il·lustrar un exemple d'entrenament d'un model neuronal modern, adjuntada en el codi font juntament amb la memòria.

2 Probabilitats

Una de les formes més senzilles d'entendre la traducció automàtica és imaginar-se que tenim una frase d'entrada A en un idioma concret, que volem traduir a una sèrie de frases possibles incloses en un conjunt de frases B d'un altre idioma. Una manera de saber quina de les frases de B és la que estem buscant, seria assignar una probabilitat a cada possible frase , i escollir-ne la de major valor, és a dir, la "més probable". Tot i que aquesta no és la manera en la qual es procedeix realment, és tracta d'una noció bàsica que ens permet entendre la importància de la probabilitat en l'àmbit de la traducció. És per aquest motiu que aquesta secció sobre probabilitats és necessària, per a donar definicions formals de què vol dir "més probable".

2.1 Definició formal d'espai de probabilitats

Definició 2.1. *Definim un experiment com un procés que és observat per tal de determinarne un resultat.*

Definició 2.2. *Definim Ω , l'espai mostral d'un experiment, com el conjunt de possibles observacions o resultats que es desprenen d'aquest.*

Definició 2.3. *Definim $A \subset \Omega$ com un esdeveniment de Ω*

Per exemple, suposem que el nostre experiment consisteix en tirar dos daus de sis cares. Aleshores

$$\Omega = \{(i, j) \mid i, j \in \mathbb{N}\,; 1 \leq i \leq 6; 1 \leq j \leq 6\}$$

On i indica el nombre obtingut al tirar el primer dau, i j indica el nombre obtingut al tirar el segon dau. Aleshores, $A = \{(2, 3), (3, 2)\}$ seria l'esdeveniment de Ω corresponent a treure un 2 i un 3 després de tirar els daus.

Definició 2.4. *Definim l'espai d'esdeveniments F com el conjunt potència d' Ω , és a dir, $F = \mathcal{P}(\Omega)$*

Definició 2.5. *Definim una funció de probabilitat o distribució de probabilitat, com una funció $P : F \to [0, 1]$ complint les següents característiques :*

- $P(\Omega) = 1$
- *Per conjunts disjunts $A_1, A_2, ...A_n \in F$, es compleix*

$$\sum_{i=1}^{n} P(A_i) = P(\bigcup_{i=1}^{n} A_i)$$

Per notar la probabilitat d'un esdeveniment A, escrivim $P(A)$.
Intuïtivament, aquestes dues propietats estableixen que donat un conjunt de possibles esdeveniments, amb seguretat ha de succeir un d'ells ; i que donat un esdeveniment format per diferents possibilitats, aquest ha de ser tan probable com la suma de les probabilitats de cada una d'elles.

Definició 2.6. *Definim un espai de probabilitats, com una terna* (Ω, F, P). *És a dir, un conjunt de possibles resultats, tots els subconjunts d'aquests possibles resultats , i una manera de quantificar quant sovint pot donar-se cada un d'aquests.*

Un es pot preguntar perquè treballem amb el conjunt potència de Ω. Això és perquè molts cops no només ens interessa saber quan pot passar un determinat esdeveniment, sinó també quan possible és una combinació d'aquests, o quan pot ser que succeeixi un però un altre no ho faci.

2.2 Independència i probabilitats condicionades

En els següents apartats, es pot considerar que ja estem treballant sota un espai de probabilitats definit. Sovint disposem d'informació sobre part del resultat d'un experiment, de manera que les probabilitats queden alterades pel que ja sabem. Per exemple, la probabilitat de treure 2 cares al tirar una moneda dos cops és impossible si en la primera tirada ja hem obtingut una creu, però no si en la primera tirada ha sortit cara. És per aquest motiu que introduïm el concepte de probabilitat condicionada:

Definició 2.7. $P(A|B)$ *o la probabilitat d'un esdeveniment A donat que l'esdeveniment B ha succeït és :*

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \tag{2.1}$$

Notis que aquesta expressió és només vàlida si $P(B) > 0$, però podem reescriure-la per admetre el cas $P(B) = 0$ de la següent manera :

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A) \tag{2.2}$$

Observis que l'última igualtat ve donada pel fet que $A \cap B = B \cap A$.
Notis que (2.2) ens permet generalitzar la possibilitat que succeeixin n esdeveniments a l'hora.
Siguin $A_1, A_2, ..., A_n$ possibles esdeveniments, aleshores:

$$P(\bigcap_{i=1}^{n} A_i) = P(A_1)P(A_2|A_1)P(A_3|A_1 \cap A_2)...P(A_n|\bigcap_{i=1}^{n-1} A_i) \tag{2.3}$$

O amb notació alternativa:

$$P(\bigcap_{i=1}^{n} A_i) = P(A_1)\prod_{j=2}^{n} P(A_j|\bigcap_{k=1}^{j-1} A_k) \tag{2.4}$$

Definició 2.8. *Diem que dos esdeveniments A,B són independents si* $P(A \cap B) = P(A)P(B)$

Observis que per (2.2), això és equivalent a què $P(A|B) = P(A)$.
Naturalment, que dos fets siguin independents és el mateix que dir que la realització d'un no afecta les probabilitats que succeeixi l'altre.

2.3 Teorema de Bayes

El Teorema de Bayes ens permet reordenar els esdeveniments per tal de calcular probabilitats condicionades. D'aquesta manera, podem calcular una en funció de l'altra, i així podem escollir aquella amb que resulti més senzill treballar. L'obtenció d'aquest és directa a partir de (2.1) i (2.2) :

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2.5}$$

Podem obtenir una versió més elaborada d'aquest teorema si considerem una família de conjunts B_i disjunts entre si, tals que $A \subseteq \bigcup_{i\in I} B_i$. Sota aquestes condicions, tenim que $A = \bigcup_{i\in I} A \cap B_i$, i donat que P és una funció de probabilitat, es compleix aleshores que

$$P(A) = P(\bigcup_{i\in I} A \cap B_i) = \sum_{i\in I} P(A \cap B_i) \tag{2.6}$$

Per la igualtat prèvia i (2.2), obtenim:

$$P(A) = \sum_{i\in I} P(A|B_i)P(B_i) \tag{2.7}$$

Així doncs, podem reformular el teorema de Bayes:

Teorema 2.1 (Teorema de Bayes)**.** *Sigui $A \subseteq \bigcup_{i\in I} B_i$, $P(A) > 0$,$B_i \cap B_j = \emptyset$ per $i \neq j$. Aleshores :*

$$P(B_j|A) = \frac{P(A|B_j)P(B_j)}{\sum_{i\in I} P(A|B_i)P(B_i)} \tag{2.8}$$

2.4 Variables aleatòries

Les variables aleatòries ens permeten poder treballar amb valors numèrics de $\mathbb{R}^n$ en lloc d'elements d'espais d'esdeveniments que hem de redefinir segons el problema.

Definició 2.9. *Una variable aleatòria X és una funció $X : \Omega \to \mathbb{R}^n$*

Definició 2.10. *Una variable aleatòria discreta és una funció X: $\Omega \to S$ on S és un subconjunt comptable de $\mathbb{R}$.*

Definició 2.11. *Una variable aleatòria contínua és una funció X: $\Omega \to T$ on T és un subconjunt no comptable de $\mathbb{R}$*

Definició 2.12. *En el cas d'una variable aleatòria discreta on $S = \{0, 1\}$, diem que es tracta d'una variable aleatòria indicador o d'un assaig de Bernoulli.*

Així doncs, quan treballem amb variables aleatòries, estudiem les probabilitats que la variable aleatòria prengui un valor determinat.

Definició 2.13. *La funció de massa de probabilitats (abreujat com pmf en anglès) d'una variable aleatòria discreta X és la funció que associa a un valor x la probabilitat que X l'assumeixi. Altrament, pmf $p(x) = p(X = x) = P(A_x)$, on $A_x = \{w \in \Omega | X(w) = x\}$*

Quan notem $X \sim p(x)$ estem indicant que la variable aleatòria discreta X està distribuïda segons la funció pmf $p(x)$

Esperança i variància

Definició 2.14. *Donada una variable aleatòria discreta X, complint* $\sum_{x\in X} p(x)|x| < \infty$ *,definim la seva esperança E(x) com:*

$$E(X) = \sum_{x\in X} p(x)x \tag{2.9}$$

Notis que $x \in X$ és un petit abús de notació, el que volem indicar és el conjunt de tots els valors x que pot prendre la variable aleatòria X.
Podem entendre l'esperança com una manera d'introduir un concepte similar a la mitjana d'un conjunt de dades en les variables aleatòries. Algunes propietats recurrents de l'esperança són:

- Si $a \in \mathbb{R}, E(aX) = aE(X)$
- Si X, Y són variables aleatòries discretes, aleshores $E(X + Y) = E(X) + E(Y)$
- Si X, Y són independents, aleshores $E(XY) = E(X)E(Y)$

En l'última propietat hem introduït el concepte de variables aleatòries independents. Entenent que les variables aleatòries són una "abstracció"d'esdeveniments d'un espai de probabilitats, podem extendre la definició d'esdeveniments independents en aquestes.

Definició 2.15. *Diem que X,Y són variables aleatòries independents si*

$$P(X = x, Y = y) = P(X = x)P(Y = y)$$

Expressat d'una altra forma, que Y prengui el valor y no altera la probabilitat que X prengui el valor x i viceversa.

Passem ara a parlar de la variància d'una variable aleatòria, que és una manera de mesurar quan consistents són els valors que pren en diversos assaigs.

Definició 2.16. *Si X és una variable aleatòria, definim la seva variància com*

$$Var(X) = E((X - E(X))^2) = E(X^2) - E(X)^2 \tag{2.10}$$

Observis que l'última igualtat en l'anterior definició és conseqüència de les propietats de l'esperança enunciades anteriorment, ja que:

$$\begin{aligned} Var(X) &= E((X - E(X))^2) = E(X^2 - 2XE(X) + E(X)^2) \\ &= E(X^2) - 2E(X)E(X) + E(E(X)^2) = E(X^2) - E(X)^2 \end{aligned}$$

Distribucions conjuntes i condicionals

Sovint un no treballa amb una única variable aleatòria, sinó amb diverses definides sobre el mateix espai de probabilitats. És sota aquestes condicions que ens cal estudiar com interactuen entre elles.

Definició 2.17. *Donades dues variables aleatòries discretes X,Y, la funció de massa de probabilitat conjunta d'ambdues es defineix com*

$$p(x, y) = P(X = x, Y = y)$$

És a dir, és la probabilitat que X prengui el valor x i alhora Y prengui el valor y.

Definició 2.18. *Donada una funció de massa de probabilitats conjunta de les variables aleatòries discretes X,Y ; definim la funció de massa de probabilitats marginal (o pmf marginal) de la variable X com :*

$$p_X(x, y) = \sum_{y \in Y} p(x, y) \tag{2.11}$$

Anàlogament, definim la funció de massa marginal de la variable Y com:

$$p_Y(x, y) = \sum_{x \in X} p(x, y)$$

2.5 Distribucions estàndards

El problema principal dels estudis probabilístics en diversos camps, especialment en el llenguatge, radica en poder trobar la funció de distribució de probabilitats P associada a la situació en concret que s'està estudiant. Així doncs, en la pràctica succeeix que mitjançant l'estudi de mostres de dades un acaba obtenint una "estimació"de P. Més endavant, s'entrarà en detall en quines tècniques són emprades per poder obtenir aquestes estimacions de les funcions.

En aquesta secció es parlarà de diverses funcions de probabilitat que apareixen recurrentment aplicant les tècniques d'estimació esmentades en el paràgraf anterior. Al llarg dels anys, les observacions empíriques han donat constància que habitualment, la majoria d'estimacions resulten ser modificacions mitjançant constants d'una funció base.

Definició 2.19. *Anomenem família en un grup de funcions de probabilitat similars entre si que difereixen només en els valors que prenen una sèrie de constants.*

Definició 2.20. *Anomenem paràmetres a les constants que diferencien cada un dels membres d'una família de funcions de probabilitat.*

En aquesta secció no només ens restringirem a les variables aleatòries discretes, sinó que també inclourem les contínues, per donar a conèixer dues de les famílies més comunes de distribució de probabilitats.

Distribució binomial

Les distribucions binomials apareixen quan un realitza una sèrie d'assaigs o proves on només existeixen dos resultats possibles (assaigs de Bernoulli) i cada un d'aquests és independent del resultat obtingut en l'anterior. Un dels exemples més clars d'aquesta distribució és el de llençar n cops una moneda per veure si cau en cara o en creu.

De manera general, aquesta família de distribucions apareix en situacions en les que un avalua l'absència o presència d'un element, el mostrar una característica concreta o no

mostrar-la ... Bàsicament, en qualsevol situació on s'estigui observant si succeeix quelcom o no (és a dir, es redueixen els resultats possibles en un esdeveniment concret i el seu complementari). És aquesta naturalesa tan simple el motiu pel qual es tracta d'una de les distribucions més comunes.

Com a comentari previ, en la notació de les famílies de probabilitats, es solen escriure primer els arguments relacionats amb la variable aleatòria que són separats dels paràmetres de la distribució amb el símbol ";".

Definició 2.21. *Definim la família de distribucions binomials, com aquelles funcions de massa probabilitat relacionades amb el fet d'obtenir r èxits després de realitzar n assaigs amb una probabilitat individual de cada èxit p:*

$$b(r; n, p) = \binom{n}{r} p^r (1-p)^{n-r} \tag{2.12}$$

Aquesta expressió prové del fet que la probabilitat de cada resultat on hagi hagut r èxits té probabilitat $p^r(1-p)^{n-r}$, i tenim en total $\binom{n}{r}$ possibles resultats complint aquestes condicions, ja que no importa l'ordre al ser els èxits i els fracassos indistingibles entre si (més en detall, si tirem una moneda, la primera creu que obtenim serà a efectes pràctics igual que la següent, per tant l'ordre és irrellevant).

En el processament del llenguatge natural, les distribucions binomials són útils per estimar probabilitats degut que a mesura que augmenta el nombre de textos que utilitzem com a base del model, algunes propietats que a priori semblen condicionades acaben tornant-se independents. Un exemple d'aquest fenomen és comptar quants cops un verb en concret és usat com a transitiu o no.

El motiu principal darrere d'aquest fet és la naturalesa mal·leable del llenguatge, que permet assignar diversos significats en una paraula o la possibilitat de trencar les normes gramaticals mantenint gran part del significat original del missatge.

Distribució normal

Com hem explicat al principi de la secció, en aquest cas no ens restringiríem només a les variables aleatòries discretes, sinó que també inclouríem les variables aleatòries contínues, i més en concret, en les funcions de densitat de probabilitat (pdf).

Cal destacar que aquestes no sempre ofereixen les millors aproximacions ja que la naturalesa d'una llengua fa que aquesta tingui tendència a freqüentment repetir paraules (com poden ser per exemple determinants o preposicions), i per tant, no podem assegurar que tots els elements tinguin una presència uniforme.

Donada aquesta distribució desigual les variables aleatòries contínues no solen ser usades per a la modelització del llenguatge i la traducció automàtica, però és necessari parlar d'elles per la seva importància en general en el camp de la probabilitat i l'estadística.

Sense entrar en detalls tècnics, ja que aquests requeririen de coneixements previs de teoria de la mesura, considerarem les funcions de densitat de probabilitat pdf, com un equivalent de les pmf però sobre les variables contínues, tot i que amb la peculiaritat que en aquest cas, la pdf en qualsevol valor concret és 0, i només pren valors diferents d'aquest en intervals. Això és pel fet que realment, la probabilitat en una variable aleatòria contínua es defineix com una integral definida de la pdf sobre un interval de valors (en

altres paraules, la probabilitat és l'àrea sota la corba definida per la pdf en un interval concret).

Definició 2.22. *Sigui X una variable aleatòria contínua, p la funció de densitat de probabilitat de X, i a, b valors possibles de X complint $a \leq b$. Definim $P(a < X < b)$ com:*

$$P(a < X < b) = \int_a^b p(x)\,dx$$

Passem ara a estudiar la distribució normal, anomenada així per la seva aparició freqüent en diversos àmbits del món real.

Definició 2.23. *Definim la família de distribucions normals, com aquelles funcions de densitat de probabilitat determinades de la següent manera:*

$$p(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{2.13}$$

3 Teoria de la Informació

El camp de la Teoria de la Informació va ser desenvolupat per Claude Shannon l'any 1940 [2]. Va néixer com a resultat d'intentar determinar quin era el màxim teòric d'informació que es podia enviar a través d'un canal de comunicació imperfecte, és a dir, que podia presentar errors en les dades enviades.

Per determinar aquestes propietats, Shannon va treballar en trobar els màxims teòrics de dues característiques fonamentals en un acte comunicatiu:

- La compressió de dades màxima : És a dir, quin és la llargada mínima de missatge per poder transmetre qualsevol contingut que desitgem.
- La ràtio de transmissió : Determinada per la capacitat del canal, que indica la "velocitat"màxima, és a dir, el màxim de dades que podem enviar a través del canal en una quantitat de temps determinat.

Si tenim en compte que al final un llenguatge és només un codi emprat per a dur a terme un procés comunicatiu, és comprensible poder aplicar els conceptes de la teoria de la informació a la lingüística, i per tant, són crucials en el camp de la traducció automàtica.

Passem doncs a estudiar algunes propietats cabdals en la teoria de la informació, i com podem aplicar aquestes al camp de la lingüística.

3.1 Entropia

Definició 3.1. *Sigui p la funció de massa de probabilitat d'una variable aleatòria discreta X que pren valors en un conjunt de símbols finit o alfabet representat per χ. Definim l'entropia de la variable aleatòria X com:*

$$H(X) = -\sum_{x \in X} p(x)\ log_2\ p(x) \tag{3.1}$$

[2] Referencia justificant això

Intuïtivament, podem entendre l'entropia de la variable X com la "quantitat d'informació"que aporta aquesta. Es per aquest motiu que usem el logaritme en base 2, ja que generalment s'expressa en bits.

No obstant, és possible expressar-ho en unitats diferents prenent una base diferent per al logaritme. En general, a partir d'ara obviarem aquesta última i simplement notarem l'entropia amb el símbol "log".

Així doncs, un pot entendre l'entropia com el nombre de bits que farien falta per representar un dels possibles resultats que prengués la variable aleatòria, d'aquí la seva relació amb la "quantitat d'informació"o la ïncertesa". Bàsicament, com més resultats diferents puguem obtindre (o vist d'una altra manera, com més "aleatorietat"hi hagi en el resultat final), més bits ens seran necessaris per representar-los.

Recordant que podem denotar la pmf d'una variable X per $p(X)$, podem redefinir l'entropia com una esperança:

$$H(X) = E(-log(p(X)) \tag{3.2}$$

Això és possible per dos fets :

- Donat que X és una variable aleatòria i $p(X)$ envia cada element de x a l'interval [0,1] tenim que $p(X)$ és per definició una variable aleatòria, ja que al final $p(X)$ envia un esdeveniment del nostre espai de probabilitats a un subconjunt de $\mathbb{R}$.
- $p(log(p(x)) = p(x)$ ja que log és una funció injectiva.

És a dir, podem entendre l'entropia com una mitjana de la quantitat d'unitats necessàries per a representar la informació d'un resultat donat per la variable aleatòria X.

Algunes propietats de l'entropia:

- $H(X) \geq 0$. Es conseqüència directa del fet que $0 \leq p(x) \leq 1$.
- $H(X) = 0$. Succeeix quan X és determinada, és a dir, només pren un únic valor amb probabilitat 1.

Entropia conjunta i condicionada

La definició de l'entropia l'associa directament a la probabilitat d'una variable aleatòria, motiu pel qual podem introduir també el concepte de l'entropia conjunta i l'entropia condicionada.

Definició 3.2. *Donades dues variables aleatòries discretes X, Y amb funció de massa de probabilitat conjunta $p(x, y)$, definim l'entropia conjunta de X i Y com :*

$$H(X,Y) = -\sum_{x \in X} \sum_{y \in Y} p(x,y)\ log(p(x,y)) = E(log(p(x,y)) \tag{3.3}$$

Cal considerar que l'entropia conjunta és el nombre mitjà de bits (o unitats corresponents) que fan falta per especificar els resultats possibles d'ambdues variables, és a dir, quants bits de mitjana requerim per indicar que a la vegada que X ha pres el valor x, Y ha pres el valor y.

Definició 3.3. *Siguin X, Y variables aleatòries discretes amb $p(x, y)$ la seva funció de massa de probabilitat conjunta. Definim l'entropia de Y condicionada per X com:*

$$\begin{aligned} H(Y|X) &= \sum_{x\in X} p(x)H(Y|X=x) \\ &= \sum_{x\in X} p(x)(-\sum_{y\in Y} p(y|x)log(p(y|x))) \\ &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)log(p(y|x)) \end{aligned} \tag{3.4}$$

Podem entendre l'entropia condicionada com la quantitat de bits extra que fan falta per expressar el valor pres per Y donat que el receptor ja coneix el resultat d' X. A partir de les probabilitats condicionades, podem generalitzar l'entropia de n variables aleatòries. Comencem reduint-nos al cas de dues variables aleatòries:

$$\begin{aligned} H(X,Y) &= -\sum_{x\in x}\sum_{y\in Y} p(x,y)\ \log p(x,y) = \\ &= -\sum_{x\in x}\sum_{y\in Y} p(x,y)\ \log p(x)p(y|x) \\ &= -\sum_{x\in x}\sum_{y\in Y} p(x,y)\ (\log p(x) + \log p(y|x)) \\ &= -\sum_{x\in x}\sum_{y\in Y} p(x,y)\log p(x) - \sum_{x\in x}\sum_{y\in Y} p(x,y)\log p(y|x) \\ &= -\sum_{x\in x}\log p(x)\sum_{y\in Y} p(x)p(y|x) + H(Y|X) \\ &= -\sum_{x\in x}\log p(x)p(x) + H(Y|X) \\ &= H(X) + H(Y|X) \end{aligned} \tag{3.5}$$

A partir d'això podem obtenir una expressió general per a l'entropia de n variables aleatòries conjuntes com:

$$H(X_1, ..., X_n) = H(X_1) + H(X_2|X_1) + ...H(X_n|X_{n-1}, ..., X_1) \tag{3.6}$$

Ràtio d'entropia

De manera general, quan parlem de lingüística, per determinar l'entropia d'un missatge solem parlar de l'entropia per lletra o l'entropia per paraula, ja que la informació i incertesa associada al missatge dependran de la longitud d'aquest.

Definició 3.4. *Per un missatge de n paraules/lletres definim la ràtio d'entropia o entropia per paraula/lletra de la següent manera:*

$$H_{rate} = \frac{1}{n}H(X_1, ..., X_n) \tag{3.7}$$

És a dir, la ràtio d'entropia és una mitjana de l'entropia condicionada de n variables aleatòries, on cada n variable aleatòria representa la possibilitat de que una lletra prengui

un símbol determinat d'un abecedari (en el cas de l'entropia per lletra) o una paraula de les existents en l'idioma (en el cas de l'entropia per paraula).

Donat que tenim un rati per a cada paraula o lletra d'un llenguatge, un pot preguntar-se si podem arribar a generalitzar la ràtio d'entropia en un llenguatge sencer. La resposta és afirmativa, però abans d'introduir la definició, hem d'introduir el concepte de procés estocàstic:

Definició 3.5. *Definim un procés estocàstic, com una família de variables aleatòries indexades* $(X_t), t \in \mathbb{N}$*, on generalment cada índex està associat a un instant de temps.*

L'exemple més senzill de procés estocàstic és possiblement el procés de Bernoulli, que consisteix en una família de n variables aleatòries de Bernoulli independents i idènticament distribuïdes, on cada variable pren el valor 1 amb una probabilitat p, i el valor 0 amb una probabilitat $1 - p$. Expressat d'una altra manera, un procés de Bernoulli és la repetició de n assaigs de Bernoulli un després de l'altre.

Introduït el concepte de procés estocàstic, podem aleshores modelar un llenguatge com un procés estocàstic format per una família de "símbols"on els símbols són paraules pertanyents a l'idioma que estem estudiant. Conseqüentment, podem introduir la ràtio d'entropia del llenguatge:

Definició 3.6. *Donat un llenguatge* $L = (X_1, ...X_n)$ *, definim la ràtio d'entropia del llenguatge com :*

$$H_{rate}(L) = \lim_{n\to\infty} \frac{1}{n} H(X_1, ..., X_n) \tag{3.8}$$

És a dir, prenem la ràtio d'entropia del llenguatge com el límit de la ràtio d'entropia d'una mostra del llenguatge a mesura que aquesta es va fent més gran. La idea principal darrere d'això és que si un va deixant estendre el nombre de símbols fins a l'infinit, s'acabarà cobrint tots els usos del llenguatge estudiat. Per entendre-ho millor, un exemple de L·L tendint a infinit"podria ser el conjunt de totes les paraules que ha dit una persona en la seva vida en ordre cronològic.

Informació mútua

Com s'ha explicat anteriorment, l'entropia condicionada ens determina la quantitat de bits extra d'informació necessaris per expressar que la variable Y ha pres un valor determinat un cop un ja coneix el valor d'una variable X. Això ens dóna a entendre que si un receptor ja coneix el valor de X, la incertesa associada a Y es veu alterada, de manera que podem dir que cada una de les variables conté part d'informació de l'altra.

Abans de donar una definició exacta d'informació mútua, cal recordar que l'entropia conjunta (3.3) és commutativa, al ser-ho les probabilitats conjuntes, i per tant, podem veure que es compleix la següent relació:

$$H(X, Y) = H(X) + H(Y|X) = H(Y) + H(X|Y) = H(Y, X) \tag{3.9}$$

D'aquesta expressió podem inferir que:

$$H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{3.10}$$

I finalment, podem donar una definició de la informació mútua:

Definició 3.7. *Siguin X, Y variables aleatòries. Definim la informació mútua entre X i Y com:*

$$I(X;Y) = H(X) - H(X|Y) \tag{3.11}$$

Notis que en la notació fem servir ";"per separar les dues variables.

Desenvolupant la definició de la informació mútua podem obtenir una expressió per aquesta a partir de les funcions de probabilitat de cada variable aleatòria:

$$\begin{aligned}
I(X;Y) &= H(X) - H(X|Y) \\
&= H(X) + H(Y) - H(X,Y) \\
&= \sum_{x\in X} p(x) \log \frac{1}{p(x)} + \sum_{y\in Y} p(y) \log \frac{1}{p(y)} + \sum_{x\in X}\sum_{y\in Y} p(x,y)\ \log p(x,y) \\
&= \sum_{x\in X}\sum_{y\in Y} p(x)p(x|y) \log \frac{1}{p(x)} + \sum_{x\in X}\sum_{y\in Y} p(y)p(y|x) \log \frac{1}{p(y)} + \sum_{x\in X}\sum_{y\in Y} p(x,y)\ \log p(x,y) \\
&= \sum_{x\in X}\sum_{y\in Y} p(x,y) \log \frac{1}{p(x)} + p(x,y) \log \frac{1}{p(y)} + p(x,y)\ \log p(x,y) \\
&= \sum_{x\in X}\sum_{y\in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}
\end{aligned} \tag{3.12}$$

A partir d'aquesta igualtat, podem enunciar diverses propietats sobre la informació mútua que ens permetran entendre millor la seva naturalesa:

- I(X;Y) és simètrica i no negativa
- $I(X;Y) = 0 \implies X, Y$ són independents
- $I(X;X) = H(X) - H(X|X)$ [3] $= H(X)$

A partir d'aquestes propietats, podem entendre la informació mútua com una manera de mesurar la dependència entre dues variables (vist d'una altra manera, com més informació ens doni X sobre Y, menys necessari serà Y per entendre la informació conjunta donada per les dues variables).

Cal afegir també que per la última propietat, en alguns casos, es pot arribar a anomenar l'entropia d'una variable X com "autoinformació".

3.2 Model *Noisy Channel*

A partir dels conceptes bàsics de teoria de la informació anteriors, Shannon va modelar el procés de comunicació a través d'un canal (entès com el mitjà a través del qual s'envia el missatge, com per exemple una línia telefònica) on hi ha present soroll (és a dir, el missatge transmès pot diferir del de entrada degut a problemes en l'estructura o integritat del canal).

La idea darrere del model és senzilla : Es pretén intentar maximitzar el flux de dades (quantitat de missatges diferents transmesos per unitat de temps) i la precisió del canal

[3] $H(X|X) = 0$ per (3.4) i pel fet que $\log p(x|x) = \log 1$

(similitud entre el missatge resultant i l'original). Per tal de fer-ho, s'assumeix que les dades de sortida del canal depenen probabilísticament de les dades d'entrada d'aquest.

El problema d'aquest objectiu radica en el fet que la compressió i la redundància són antagonistes. És a dir, a més intentem limitar la llargada del missatge (i per tant maximitzem el flux), menys espai estem deixant per a repetir les dades per tal de poder ser aquestes recuperades en cas que el soroll del canal les hagi modificat (disminuint així la precisió).

En definitiva, el que s'està buscant és una manera de codificar el missatge per tal que aquest ocupi el menor espai possible, però disposi de la redundància necessària per a poder detectar errors i recuperar el missatge original després de descodificar la informació rebuda.

Un dels conceptes centrals associats a aquest model és la capacitat del canal.

Definició 3.8. *Definim la capacitat d'un canal com la màxima ràtio de transmissió de informació amb una probabilitat arbitràriament baixa de no poder recuperar el missatge original a partir del rebut.*

Abans de donar els resultats de Shannon sobre la capacitat d'un canal, necessitem introduir una definició prèvia:

Definició 3.9. *Un canal sense memòria és un canal en el qual el resultat actual només depèn de l'entrada actual, és a dir, el resultat actual és independent dels resultats i entrades anteriors.*

Shannon va establir el següent teorema sobre la capacitat d'un canal sense memòria:

Teorema 3.1 (Capacitat en canal segons informació mútua)**.** *Sigui C la capacitat màxima possible d'un canal sense memòria, X el missatge d'entrada codificat, Y el missatge de sortida codificat. Aleshores:*

$$C = \max_{p(X)} I(X;Y)$$

En altres paraules, aquest teorema de Shannon estableix que la capacitat màxima d'un canal s'assoleix quan dissenyem un codi d'entrada la distribució del qual maximitza la informació mútua entre entrada i sortida. Com a exemple, considerem un canal que com a entrada rep un bit i com a sortida, a causa del soroll, retorna el bit invertit amb una probabilitat p (i per tant, el bit original amb probabilitat $1-p$) Tenim aleshores que:

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) = H(Y) - H(p) \\ &= H(Y) - \sum_{x,y\in\{0,1\}} p(x,y)\log p(y|x) \\ &= H(Y) - (p(X=0)+p(X=1))(p\log p + (1-p)\log(1-p)) \\ &= H(Y) - (p\log p + (1-p)\log(1-p)) \\ &= H(Y) - H(p) \end{aligned} \tag{3.13}$$

Observis que donat que p es tracta d'una constant fixada, l'únic element que pot variar és $H(Y)$. Donat a més que l'entropia és no negativa, tenim que obtindrem el màxim quan $H(Y)$ tingui el màxim valor possible. Finalment, tenim que l'opció adient és fer la

distribució de la probabilitat d'entrada (i per tant la de sortida) uniforme, obtenint així una entropia màxima de 1 bit :

$$\max_{p(X)} I(X;Y) = 1 - H(p) \tag{3.14}$$

Tenim doncs que $C \leq 1$, sent $C = 1$ només quan $p = 0$ o $p = 1$. És a dir, si sabem que el bit s'inverteix (o es manté) amb seguretat, sempre serem capaços de recuperar la informació original amb total precisió. Altrament, podem realitzar estimacions aproximades, ja que si $p > \frac{1}{2}$, podem assumir que el bit original sigui l'invers del rebut; i de manera anàloga, si $p < \frac{1}{2}$ aleshores ens hauríem de decantar per interpretar el bit rebut com l'original.

Notis que en el cas que $p = \frac{1}{2}$ aleshores $H(p) = 1$, i per tant, $C = 0$. És a dir, donat que és tan probable la inversió com el manteniment del bit original, aleshores el missatge rebut no ens permet inferir res sobre el missatge enviat, motiu pel qual tenim que $I(X;Y) = C = 0$. En altres paraules, "Y no conté informació sobre X", donant lloc a un canal inútil per a la comunicació.

La importància d'aquest model per a la lingüística i la traducció automàtica radica en el fet que podem fer ús d'aquest per modelar una situació de traducció entre dos idiomes.

De manera genèrica, podem representar un model *Noisy Channel* amb el diagrama mostrat a la figura 1.

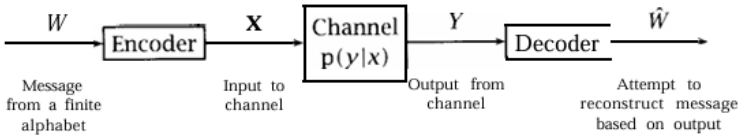

Figura 1: Representació estàndard del model Noisy Channel

Noisy Channel aplicat a la lingüística

En el cas de la lingüística, però aquest model canvia. Bàsicament, no tenim control sobre la fase de codificació ja que el codi en si és l'idioma usat per l'emissor i el missatge original era el significat que volia transmetre aquest.

Passem doncs a estar en una situació en la que el nostre objectiu és intentar descodificar l'output rebut per a intentar obtenir l'input original ja codificat. La següent figura és una representació de com seria aquest model Noisy Channel modificat, on I és el conjunt de símbols del missatge d'entrada, i és cada símbol individual del input, O és el conjunt de símbols del missatge de sortida, o cada símbol individual del output i $\hat{I}$ la nostra estimació del input més probable: Per determinar l'input més probable, fem servir el teorema de

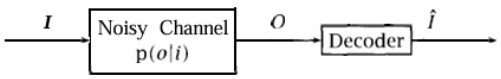

Figura 2: Model Noisy Channel per a problemes lingüístics

Bayes i considerem que la probabilitat del output és constant:

$$\hat{\mathrm{I}} = arg\max_i p(i|o) = arg\max_i \frac{p(i)p(o|i)}{p(o)} = arg\max_i p(i)p(o|i) \tag{3.15}$$

Expressat d'una altra forma, busquem cada símbol i del missatge d'entrada més probable donat que hem obtingut el símbol o en l'output. Observis que tenim dues distribucions de probabilitat diferents :

- $p(i)$: El model del llenguatge. És a dir, la probabilitat de cada seqüència de paraules en el llenguatge original.
- $p(o|i)$: La probabilitat del canal. És a dir, la probabilitat de que s'hagi obtingut la paraula de sortida donada la d'entrada.

Aplicant aquest model al problema de traducció entre dos idiomes, podem entendre que l'output és la frase en l'idioma original, i l'input és la frase en l'idioma al que hem de traduir que ha estat alterada pel soroll del canal. D'aquesta manera, el problema de traducció es resumeix en descodificar el missatge en l'idioma original per obtenir el de destí.

3.3 Entropia relativa o Divergència Kullback-Leibler

Definició 3.10. *Siguin p(x),q(x) distribucions de probabilitat discretes. Aleshores, definim l'entropia relativa o Divergència Kullback-Leibler (o Divergència-KL) entre les dues com:*

$$D(p||q) = \sum_{x\in X} p(x)\log\frac{p(x)}{q(x)} \tag{3.16}$$

Aquí ens fa falta definir que $0\log\frac{0}{q(x)} = 0$ i $p(x)\log\frac{p(x)}{0} = \infty$, per poder cobrir els valors de la divergència en tots els casos possibles.

La divergència-KL presenta dues propietats interessants:

- $D(p||q) \geq 0$
- $D(p||q) = 0 \iff p = q$

Ambdues són demostrades mitjançant la desigualtat de Gibbs:

Teorema 3.2 (Desigualtat de Gibbs)**.** *Siguin p,q funcions de distribució de probabilitats discretes de la variable aleatòria X. Aleshores:*

$$-\sum_{i=1}^{n} p(x_i)\log p(x_i) \leq -\sum_{i=1}^{n} p(x_i)\log q(x_i) \tag{3.17}$$

Complint-se la igualtat si i només si $p(x_i) = q(x_i)$ per $i = 1, ..., n$.

Demostració. Considerem $I = \{i \in N | 1 \leq i \leq n,\ p(x_i) \neq 0\}$. Aleshores, tenint en compte que $\log x \leq x - 1$ si $x > 0$ (aquest fet es pot provar analitzant les derivades de la funció $x - 1 - \log x$ i observant que té un únic mínim a $x = 1$) obtenim :

$$-\sum_{i\in I} p(x_i)\log\frac{q(x_i)}{p(x_i)} \geq -\sum_{i\in I} p(x_i)(\frac{q(x_i)}{p(x_i)} - 1) = \sum_{i\in I} p(x_i) - \sum_{i\in I} q(x_i) = 1 - \sum_{i\in I} q(x_i) \geq 0$$

On la última desigualtat ve del fet que podem haver exclòs algun i per al qual $q(x_i) \neq 0$ i per tant el sumatori dels $q(x_i)$ pot ser menor que 1.

Fins ara hem provat que sobre el nostre conjunt I tenim:

$$-\sum_{i\in I} p(x_i) \log \frac{q(x_i)}{p(x_i} \geq 0$$

I equivalentment:

$$-\sum_{i\in I} p(x_i) \log q(x_i) \geq -\sum_{i\in I} p(x_i) \log p(x_i)$$

Aquesta desigualtat es pot ampliar a tot el conjunt de n possibles valors de la variable aleatòria X, (és a dir, incloent aquells x_i per als quals $p(x_i) = 0$) al ser $\lim_{p(x)\to 0} p(x) \log p(x) = 0$ i $\lim_{q(x)\to 0} -\log q(x) = \infty$.

Obtenim així la desigualtat buscada :

$$-\sum_{i=1}^{n} p(x_i) \log q(x_i) \geq -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

Finalment, en el cas d'igualtat, s'ha de complir que

- $\sum_{i\in I} q(x_i) = 1$. És a dir, $q(x_i) = 0$ si $i \notin I$.
- $\forall i \in I$, $\frac{q(x_i)}{p(x_i)} = 1$ per assegurar que $\log \frac{q(x_i)}{p(x_i} = \frac{q(x_i)}{p(x_i)} - 1$

Aquestes condicions es donen si i només si $q(x_i) = p(x_i)$ per $i = 1, ..., n$ □

Per concloure, les propietats de la divergència-KL són immediates després de veure que podem reordenar la desigualtat de Gibbs per obtenir:

$$0 \leq \sum_{i=1}^{n} p(x_i) \log \frac{p(x_i)}{q(x_i)} \tag{3.18}$$

Les propietats de no negativitat i igualtat a 0 si i només si les dues distribucions són iguals ens poden permetre inferir que la divergència-KL pot actuar com una "distància" (formalment no és una distància, al no ser simètrica) i per tant, pot ser entesa com una forma de quantificar quan diferents són dues distribucions de probabilitats entre si.

En una interpretació més adient a la teoria de la informació, diem que l'entropia relativa és la mitjana de bits que es perden quan intentem codificar esdeveniments d'una distribució p amb un codi associat en una distribució q.

De fet, es pot redefinir la informació mútua com la divergència-KL entre una distribució conjunta de la independència dels seus termes:

$$I(X;Y) = D(p(x,y)||p(x)p(y)) \tag{3.19}$$

Per tant, és important recalcar el fet que tot i que no sigui una distància com a tal, la divergència-KL es tracta d'una manera de mesurar la diferència entre dues distribucions de probabilitat. Aquest fet serà crucial en la següent secció.

3.4 Entropia creuada o *Cross Entropy*

Durant les seccions anteriors hem anat fent petits incisos per explicar com podríem aplicar els conceptes anteriors en l'àmbit del llenguatge, però és gràcies a l'entropia relativa que podem passar a donar un marc formal per tal de treballar matemàticament amb problemes lingüístics.

Suposem que partim d'un conjunt d'expressions en un idioma determinant. Mapejant aquestes a nombres segons creiem convenient, podem representar-les mitjançant una variable aleatòria X. A continuació, assumim que existeix una distribució de massa de probabilitat associada en aquesta variable aleatòria, al saber que hi ha expressions que apareixen més sovint que d'altres. Per tant tenim $X \sim p(x)$.

El problema radica en que per a cada conjunt d'expressions en específic, donat la naturalesa del llenguatge (dependència del context, mal·leabilitat, variabilitat segons el parlant ...) mai podem donar una distribució exacta per a cada cas. No obstant, mitjançant estudis de grans quantitats d'expressions (com per exemple, en reculls de textos, extractes de conversacions...) o *corpus*, podem deduir una estimació de $p(x)$. En aquesta estimació, l'anomenarem el *model* m de la distribució p.

Per intentar obtenir el millor model possible, hauríem d'escollir aquell m que minimitzés l'entropia relativa $D(p||m)$, per tal d'obtenir l'estimació que més s'apropi a la distribució real. S'ha de tenir en compte però que aquest càlcul sovint resulta impossible ja que no coneixem el valor exacte de p.

Per solucionar aquest problema, introduïm el concepte d'entropia relativa:

Definició 3.11. *Sigui X una variable aleatòria discreta amb funció de massa de probabilitat p, q una altra funció de massa de probabilitat. Definim l'entropia creuada o cross entropy entre X i q com :*

$$H(X,q) = H(X) + D(p||q) \tag{3.20}$$

Notis que a partir de la definició d'entropia i divergència-KL, podem obtenir l'expressió equivalent següent per a l'entropia creuada:

$$H(X,q) = H(X) + D(p||q) = -\sum_{x\in X} p(x)\log p(x) + \sum_{x\in X} p(x)\log\frac{p(x)}{q(x)} = -\sum_{x\in X} p(x)\log q(x) \tag{3.21}$$

De la mateixa forma en que vam definir l'entropia d'un llenguatge a (3.8), podem definir també l'entropia creuada d'un llenguatge L respecte d'un model m d'aquest:

Definició 3.12. *Sigui $L = (X_1, ..., X_n) \sim p(x)$ un llenguatge, m un model de p, $S = \{(x_1, ..., x_n)\,|x_i \in X_i, i = 1, ..., n\}$ el conjunt de totes les seqüències possibles del llenguatge. Definim l'entropia creuada de L respecte del model m com:*

$$H(L,m) = \lim_{n\to\infty} -\frac{1}{n}\sum_{s\in S} p(s)\log m(s) \tag{3.22}$$

Tinguis en compte que $p(s)$ i $m(s)$ representen la probabilitat conjunta dels components de s.

És a dir si $s = (a_1, ..., a_n)$ aleshores $p(s) = p(a_1, ..., a_n)$. A primera vista, aquestes definicions no semblen útils, ja que requereixen conèixer la distribució exacta del llenguatge o de la variable aleatòria discreta.

No obstant, podem assumir diverses condicions per aconseguir una aproximació millor, no depenent de la probabilitat real de cada seqüència:

$$H(L,m) = -\lim_{n\to\infty} \frac{1}{n} \log m(s) \tag{3.23}$$

En aquest cas, aquesta simplificació es pot acceptar si considerem que el procés és estacionari i ergòdic.

Definició 3.13. *Diem que un procés estocàstic $S = (X_1, ..., X_n)$ és estacionari si $\forall l, k \in \mathbb{N}$ es compleix que :*

$$P(X_1 = x_1, X_2 = x_2, ...X_k = x_k) = P(X_{1+l} = x_1, X_{2+l} = x_2, ..., X_{k+l} = x_k) \tag{3.24}$$

De manera intuïtiva, un procés estocàstic estacionari és aquell que no canvia en el temps. Independentment de quantes observacions fem, la probabilitat dels esdeveniments en cada una d'aquestes seguirà sent la mateixa. Vist d'una altra forma, es podria dir que en un procés estacionari el pas del temps no afecta la probabilitat dels resultats.

Definició 3.14. *Diem que un procés estocàstic $S = (X_1, ..., X_n)$ és ergòdic si per una funció mesurable f es compleix:*

$$\lim_{n\to\infty} \frac{1}{n} \sum_{t=1}^{n} f(X_t) = \mathbb{E}[f(X_t)] \tag{3.25}$$

De manera informal, diem que un procés estocàstic és ergòdic quan una observació suficientment llarga del procés és suficient per a poder determinar el comportament típic d'aquest. Dit d'una altra forma, com més gran sigui la mostra més representativa serà.

Quan tenim un procés ergòdic i estacionari, aleshores es compleix el teorema de Shannon-McMillan-Breiman o la *Asymptotic Equipartition Property* [9, pp. 645–646]:

Teorema 3.3 (Shannon-McMillan-Breiman)**.** *Sigui $S = (X_1, X_2, ..., X_n)$ un procés estocàstic finit ergòdic i estacionari. Si H és l'entropia de S, aleshores es compleix:*

$$-\frac{1}{n} \log p(X_1, \ldots, X_n) \to H \quad \textit{amb probabilitat 1}$$

És a dir, la successió anterior convergeix quasi segurament amb l'entropia H de S.

Expressat de manera diferent, el teorema ens assegura que per entendre la informació que ens està aportant un procés estocàstic ergòdic i estacionari, només ens fa falta estudiar una seqüència "prou llarga"d'aquest.

Retornem ara a (3.22). Observis que l'expressió és equivalent a calcular el límit de l'esperança

$$\mathbb{E}_p(\frac{1}{\log m(s)})$$

, que no és sinó una mitjana ponderada. La idea és que a partir del teorema anterior, podem entendre que si la seqüència a partir de la qual calculem l'entropia creuada del llenguatge és suficientment llarga, no necessitem fer aquesta ponderació, ja que estarem observant el comportament típic del procés i no ens serà necessari avaluar totes les possibles seqüències, obtenint així l'expressió simplificada (3.21).

Per tant, la versió simplificada s'ha d'entendre més com una conseqüència çonceptual"del teorema aplicada a l'estudi de l'entropia creuada d'un llenguatge i el seu model.

Hom pot pensar que aquest raonament només és vàlid si el llenguatge és un procés estocàstic estacionari i ergòdic. Realment, un llenguatge no és estacionari, degut que amb el temps apareixen noves paraules i canvia la popularitat d'algunes expressions.

No obstant, s'ha de tenir en compte l'existència d'estàndards per als llenguatges i que la naturalesa dels canvis lingüístics és gradual i no sobtada.

Tenint en compte aquestes últimes característiques, en la pràctica es pot considerar un llenguatge com un procés estacionari, ja que generalment els corpus seleccionats pel modelatge són coherents entre si i presenten una versió cohesionada d'aquest, evitant les diferències temporals significatives.

4 Modelització estadística del llenguatge

Un cop establertes les bases del nostre treball, procedim a investigar i detallar diverses maneres possibles d'obtenir les aproximacions de les distribucions de probabilitat reals d'un llenguatge, és a dir, aprofundirem en com realment es creen els models.

Recordem abans de començar que les tasques de modelització del llenguatge són aquelles que consisteixen en predir la següent paraula d'una seqüència donades les anteriors. En principi sembla una tasca senzilla, però la immensitat del vocabulari d'una única llengua, les normes gramaticals i la dependència contextual dificulten l'obtenció d'una resposta definitiva. Per això, al llarg del temps es van anar desenvolupant diverses tècniques de modelatge, cada una basada en un aspecte del llenguatge.

En aquest cas, s'ha de veure que el nostre objectiu és predir una característica objectiu a partir d'una sèrie de característiques classificatòries. Aquest fet es pot entendre com compartimentar el nostre conjunt de dades mitjançant classes d'equivalència, i fer ús d'aquestes per tal de determinar el valor de la nostra característica objectiu en dades futures. És necessari destacar que aquest mètode presenta en si una assumpció implícita d'independència de les dades respecte dels trets que no pertanyen a les classes d'equivalència definides com a classificatòries, el qual porta a un comportament limitant, i en certa part "discriminatori", dels models.

Així doncs, una de les condicions més importants en l'elaboració de models és intentar triar un nombre adient de trets classificatoris, buscant un equilibri entre capacitat de discriminació al augmentar el nombre de classes i de fidelitat al disminuir el nombre de classes. Més en detall, l'augmentar el nombre de classes ens permet obtenir més informació al disposar de més característiques que comparar, però augmentar el nombre en excés pot fer que moltes d'aquestes acabin amb una quantitat insignificant d'informació que realment no ens permetrà fer estimacions significatives. Per altra banda, En resum, un ha de poder trobar un equilibri entre "filar massa prim i no filar prou".

Per altra banda, també és important destacar que les dades d'entrenament (és a dir, per a la creació del model) són essencials, i per tant, cal fer una selecció acurada d'aquestes. És obvi que per aquesta tasca farem reculls de grans quantitats de textos extrets d'una o diverses fonts (anomenats *corpus* en singular i *corpora* en plural), per tal de poder obtenir la visió més representativa possible de la llengua a ser estudiada. A nivell de curiositat, moltes institucions lingüístiques solen posar a disposició corpus seleccionats pels seus

acadèmics amb textos de diversos estils, temàtiques, períodes i autors per tal de facilitar l'accés a l'estudi de les seves corresponents llengües. Per exemple, localment, disposem del CTILC (Corpus textual informatitzat de la llengua catalana) [8] proporcionat per l'IEC i el Corpus del Español del Siglo XXI (CORPES XXI) [12] proporcionat per la RAE.

Tenint en compte tot això, ara procedirem a investigar diverses tècniques per obtenir un model d'un llenguatge determinat.

4.1 Modelització per n-grames

Es tracta de la modelització més senzilla i intuïtiva possible. Bàsicament, consisteix a determinar la probabilitat de la pròxima paraula a partir de l'estat de les $n-1$ anteriors. En general, podem dir que estem fent una *assumpció de Màrkov*[18, p.34], ja que considerem que només l'estat actual (en aquest cas, la paraula anterior) determina l'estat futur (és a dir, la pròxima paraula).

De forma lògica, podem augmentar el nombre de paraules que determinen el nostre estat actual, tenint així en compte les dos, tres, quatre, o en general les $n-1$ paraules anteriors. És a dir, si x_m és la m-èssima paraula dins el text considerem que:

$$\begin{aligned} &P(X_m = x_m \mid X_{m-1} = x_{m-1}, \ldots, X_1 = x_1) = \\ &P(X_m = x_m \mid X_{m-1} = x_{m-1}, \ldots, X_{m-n} = x_{m-n+1}) \end{aligned} \tag{4.1}$$

On n indica l'ordre del nostre model. Notis que tot i que només considerem les $n-1$ paraules anteriors, la nomenclatura considera també la paraula actual.

Per a certs casos, existeixen nomenclatures concretes. Així per un model que només té en compte la paraula anterior, l'anomenem un model per *bigrames* i un que considera les dues anteriors és anomenat un model per *trigrames*.

Dintre d'aquests models, considerem com a classes d'equivalència cada un dels conjunts de $n-1$ paraules anteriors. És senzill veure que aquesta estructura pot no ser la millor, perquè no aporta realment significat sobre el context, al només fixar-te en els elements més pròxims. Aquest fet és especialment notable quan estem considerant el cas dels bigrames, que només es redueixen a fixar-se en la paraula anterior.

La inconveniència anterior pot ser mitigada incrementant n, però el conjunt de classes d'equivalència possibles creix exponencialment segons aquest paràmetre. Si V és el nombre de paraules totals que poden aparèixer (és a dir, el nostre vocabulari), $N(n)$ el nombre de classes d'equivalència segons l'ordre n del model i considerem que en les $n-1$ paraules anteriors no hi ha cap repetició:

$$N(n) = \prod_{i=0}^{n-1} V - i \leq \prod_{i=1}^{n} V = V^n \implies N(n) \in O(V^n) \tag{4.2}$$

Per tant, ràpidament deixa de ser viable a nivell computacional aplicar-lo per a valors alts. Tornem doncs en una situació on s'han d'equilibrar complexitat i utilitat, que en la pràctica es soluciona generalment prenent un model per trigrames.

Les mancances d'aquest model són evidents, i existeixen altres maneres d'obtenir classes d'equivalència que consisteixen en no treballar directament amb les paraules anteriors,

sinó amb altres elements. Per exemple, es pot treballar amb els lexemes de les paraules o agrupar les paraules en grups semàntics segons el seu significat.

No obstant, la introducció d'aquests també implica un augment de la complexitat, ja que requereixen d'un preprocessament del text o de l'existència de classificacions prèvies de les paraules del vocabulari, motiu pel qual es sol preferir la modelització per n-grames a causa de la seva senzillesa.

Un cop definides les nostres classes d'equivalència mitjançant n-grames ens cal passar a la part complicada i escollir un bon estimador per a la probabilitat de la següent paraula.

Estimador de màxima versemblança (*MLE*)

Per exemplificar aquest estimador, considerem que estem modelant un llenguatge per trigrames a partir d'un corpus en anglès, i a partir de les nostres dades, observem que tenim en 10 ocasions l'aparició de les paraules prèvies *comes across*. D'aquests 10 casos, en 8 la següent paraula és *as*, en 1 és *more* i en l'últim és *a*. Donades aquestes dades, un primerament ha de pensar com establir una funció matemàtica que a partir del paràmetre d'entrada *comes across*, ens permeti determinar la probabilitat de la següent paraula.

En la modalització per n-grames, el paràmetre del que estem parlant no és en si un únic valor, sinó que és el conjunt de valors determinat per les probabilitats condicionades de que la paraula w pertanyent al nostre vocabulari aparegui després de les "n-1"paraules anteriors.

En general, podem expressar quan bé s'adapta un paràmetre que nosaltres donem a les dades observades mitjançant l'anomenada funció de versemblança [7, p.290]:

Definició 4.1. *Sigui $f(x|\theta)$ la funció de massa de probabilitat conjunta de la nostra mostra $X = (X_1, ..., X_n)$. Aleshores, considerant que les dades observades en la mostra són $x = (x_1, ..., x_n)$, definim la funció de versemblança de θ com :*

$$L(\theta|x) = f(x|\theta)$$

Així doncs, podem prendre com una primera estimació de les probabilitats de la futura paraula a partir de maximitzar la funció de versemblança. Bàsicament, busquem el conjunt de probabilitats condicionades que s'ajustin millor a la mostra observada per cada classe d'equivalència formada per les $n-1$ paraules anteriors. En aquest cas, trobar aquest conjunt de probabilitats és trobar l'estimador de màxima versemblança:

Definició 4.2. *Per en una funció de versemblança $L(\theta|x)$ de la mostra observada, definim l'estimador de màxima versemblança $\hat{\theta}$ com :*

$$\hat{\theta} = \underset{\theta}{argmax}\ L(\theta|x)$$

En el cas de la modelització per n-grames, el nostre estimador de versemblança serà literalment el conjunt de probabilitats condicionades obtingudes en l'observació de les dades de la nostra mostra. És a dir, prenem una mentalitat freqüentista i assumim que les futures dades que nosaltres observarem es comportaran de la mateixa forma que les que ja hem estudiat. Per tant, la probabilitat de la paraula següent en cada n-grama vindrà donada per:

$$P_{MLE}(w_n|w_1, ..., w_{n-1}) = \frac{C(w_1, ..., w_n)}{C(w_1, ..., w_{n-1})} \tag{4.3}$$

On $C(w_1, ..., w_n)$ indica el nombre d'ocurrències de les paraules $w_1, ..., w_n$ seguides en el nostre corpus.

Basant-nos en les dades donades com a exemple a l'inici de la secció, les probabilitats obtingudes segons l'estimador de màxima versemblança serien:

$$P(as|come, across) = \frac{8}{10}$$
$$P(more|come, across) = \frac{1}{10}$$
$$P(a|come, across) = \frac{1}{10}$$
$$P(w|come, across) = 0$$

On w es refereix a qualsevol paraula del corpus que no sigui ni *a*, ni *as*, ni *more*.

En altres paraules, donar les probabilitats mitjançant el mètode de l'estimador de màxima versemblança consisteix en calcular les probabilitats relatives de cada n-grama per a cada classe d'equivalència.

Cal afegir que aquest estimador de màxima versemblança s'aconsegueix quan es pren com a base la independència entre els diversos n-grames per tal de facilitar el càlcul de les probabilitats. Aquesta assumpció és completament errònia per dos motius principals:

Primer, els n-grames es solapen entre si, de manera que l'assumpció d'independència és equivocada. Per exemplificar-ho, suposem que estem elaborant un model per trigrames de l'oració *No creu que pugui*. En aquest cas tenim dos trigrames diferents "No creu que" i "creu que pugui", que és solapen en el fragment "creu que". A partir del primer, determinaríem la probabilitat $P(que|no, creu)$, i en el segon es faria el mateix amb $P(pugui|creu, que)$. El problema radica en el fet que es considerarien completament independents quan estan compartint un context comú (entenent context com les paraules que el precedeixen) que modificaria la probabilitat del trigrama següent. En altres paraules, que el trigrama anterior sigui "no creu que" augmenta la probabilitat que el següent trigrama sigui "creu que pugui".

Segon, el fet que una paraula aparegui en un text, augmenta la probabilitat de que torni a aparèixer de nou. Aquest fet està associat a dues propietats inherents del llenguatge contrastades empíricament [19, p.198]:

- Llei de Zipf : "La freqüència d'una paraula és inversament proporcional a la seva posició en la taula de freqüències".

  Bàsicament, aquesta llei enuncia que dintre d'un corpus de text, es troba un grup de poques paraules repetides freqüentment, mentre que la majoria de paraules apareixen rarament. En aquest cas, la freqüència de la paraula fa referència al nombre total d'ocurrències d'aquesta en el corpus, i la taula de freqüències és aquella on es recullen les paraules ordenades descendentment segons la seva freqüència (és a dir, la paraula que més ocurrències presenta ocuparà la primera posició, la següent amb més aparicions el segon lloc i així successivament).

- Agrupació i recurrència sobtada de paraules: Quan un text tracta d'un tema principal, les paraules referents en aquest apareixen més sovint que en altres de diferent temàtica (agrupació). A més, quan dintre d'aquest tema en concret s'explica un concepte determinat en una secció, la paraula associada en aquest presenta una

freqüència molt elevada a nivell local (recurrència). Així doncs, la paraula "òxid" serà molt més comú en un llibre de química que no pas en una novel·la històrica; i dintre del primer trobarem aquesta mateixa paraula repetida més en les seccions dedicades als compostos que continguin oxigen que no pas al prefaci.

Per altra banda, cal elaborar sobre un dels grans problemes de l'estimador de màxima versemblança. En aquest es reparteix tota la probabilitat de les funcions de massa en els exemples observats al corpus, assignant una probabilitat nul·la a les possibles combinacions de paraules que no hagin estat incloses en els textos d'entrenament.

Una de les primeres idees per poder arribar a solucionar aquest inconvenient és incrementar la mida del nostre corpus, però això comportaria un increment del nombre de paràmetres del nostre model, el qual a la vegada portaria a una major complexitat computacional i, a la llarga, podria fer que quedés inservible. A més a més, cal tenir en compte que poder recollir un corpus on s'incloguin totes les combinacions possibles de n-grames és una tasca pràcticament impossible.

Per tant, resulta convenient canviar la nostra perspectiva, i en comptes de centrarnos en ampliar el nostre corpus, caldria trobar una altra forma d'estimar les possibilitats associades al model.

En aquest cas, una primera aproximació seria disminuir lleugerament el valor dels resultats observats i augmentar els no observats de manera que no tinguessin probabilitat nul·la. En aquest procés, se'l sol anomenar un procés de *smoothing* o "suavització". A continuació procedim a explicar en detall algunes tècniques de *smoothing* aplicades en la modelització per n-grames.

Estimador de Laplace

Una de les tècniques d'*smoothing* possibles per ser aplicades en aquest consisteix en fer ús de la llei de successió de Laplace. Introduïda com el indica el seu nom per Pierre-Simon Laplace l'any 1774 per determinar quina era la probabilitat que el sol sortís el dia següent [16, pp.563–564]; aquesta llei aconsegueix suavitzar les probabilitats partint primerament d'una assumpció d'uniformitat a priori (en altres paraules, s'assumeix que cada resultat té les mateixes probabilitats d'ocórrer) que va sent actualitzada a partir de les observacions realitzades.

Teorema 4.1 (Llei de successió de Laplace)**.** *Siguin N les observacions realitzades d'un experiment amb K possibles resultats. Si s'ha observat el resultat k un total de n vegades, aleshores la probabilitat de que en la següent observació es torni a obtenir el mateix resultat és:*

$$P_{LAP}(k) = \frac{n+1}{N+K}$$

On s'ha usat la notació $P_{LAP}(k)$ per denotar la probabilitat del succés k sota les condicions anteriors. Una prova d'aquesta llei i un enunciat més general pot ser trobada a [16, pp.568–571].

Aplicat en concret al nostre cas de modelització per n-grames, si considerem $C(w_1, ..., w_n)$ el nombre de cops que apareix l'n-grama format per les paraules $(w_1, ..., w_n)$, N el nombre d'n-grames total en el nostre corpus i B el nombre possible d'n-grames, aleshores la

probabilitat de cada n-grama $(w_1, ..., w_n)$ és :

$$P_{LAP}(w_1, ..., w_n) = \frac{C(w_1, ..., w_n) + 1}{N + B} \tag{4.4}$$

Observis que d'aquesta manera, si un n-grama concret no ha aparegut en el corpus almenys se li assignarà una probabilitat no nul·la, dotant així al model una capacitat de considerar resultats que a partir de les dades d'entrada serien "impredictibles".

Tot i així, aquest model no està lliure dels seus inconvenients, i fins i tot, és possible argumentar que donaria resultats tant poc adients com els donats per l'estimador de màxima versemblança. El problema aquí radica en que s'estan repartint totes les probabilitats entre tots els possibles n-grames del nostre vocabulari. Aquest fet implica que s'està donant massa importància a valors que en la pràctica no apareixeran mai; en altres paraules, es podria dir que estem "suavitzant" massa el model.

No resulta doncs convenient aplicar aquesta tècnica directament, ja que com s'ha comentat anteriorment, el llenguatge no tendeix a ser uniforme, sinó que hi ha certes paraules o expressions que solen aparèixer més sovint (recordis la llei de Zipf), especialment si ens centrem en aplicacions d'aquest en àmbits o contexts concrets (recordis l'efecte d'agrupació i recurrència sobtada de paraules).

Caldria doncs, trobar una forma de poder conciliar la capacitat de la llei de Laplace per poder predir resultats no trobats en el corpus amb les tendències a l'agrupació i repetició del llenguatge.

Llei de Lidstone i llei de Jeffrey-Parks

Una de les primeres solucions possibles als inconvenients presentats per l'estimador mitjançant la llei de successió de Laplace és fent ús de l'anomenada llei de Lidstone [19, p.204], on introduïm dos nous paràmetres A i λ per controlar el grau de *smoothing*:

Teorema 4.2 (Llei de successió de Lidstone)**.** *Siguin N les observacions realitzades d'un experiment amb K possibles resultats, $A \in \mathbb{R}^+$ i $\lambda \in \mathbb{R}^+$, n el nombre de vegades que s'ha observat un resultat determinat k. Aleshores:*

$$P_{Lid}(k) = \frac{n + A}{N + \lambda K}$$

Aplicat a la modelització per n-grames, obtenim que la probabilitat segons la llei de successió de Lidstone per a l'n-grama $(w_1, ..., w_n)$ és:

$$P_{Lid}(w_1, ..., w_n) = \frac{C(w_1, ..., w_n) + A}{N + \lambda B} \tag{4.5}$$

En aquest cas, es sol aplicar valors inferiors a 1 en els paràmetres A i λ, per tal de poder controlar la probabilitat que volem assignar-li a cada n-grama que no hagi aparegut i reduir el nombre total de possibles n-grames que estem considerant. D'aquesta manera aconseguim no assignar gran part del pes a valors que és molt possible que no trobem mai, assegurant així una "suavització" de la probabilitat molt més controlada i que podem adaptar a les nostres necessitats.

Observis que podem considerar la llei de Lidstone com una interpolació entre els valors de l'estimador de màxima versemblança i una distribució uniforme (és a dir, assignant

probabilitat $\frac{1}{B}$ a cada n-grama) quan $\lambda = A$. Per fer-ho, considerem $\mu = N/(N + \lambda B)$, i aleshores es compleix:

$$P_{Lid}(w_1, ..., w_n) = \mu \frac{C(w_1, ..., w_n)}{N} + (1 - \mu)\frac{1}{B} \tag{4.6}$$

No obstant, sempre es solen diferenciar els paràmetres del numerador i el denominador per tenir millor control sobre el valor que assignem a cada n-grama que no hagi aparegut, d'aquí que tractem A i λ com a valors diferents.

Alternativament, en el cas que $A = \lambda$ es sol escollir el valor $\lambda = 0.5$, principalment pel fet que "equilibra" la importància que donem a les dades observades i a una assumpció de distribució uniforme.

Estimador Held out

Hom pot observar que tots aquests mètodes usats fins ara són heurístics, és a dir, intenten predir com es comportarà el text a partir del que ja ha succeït. Generalment, a l'hora d'entrenar els models, un no només utilitza el corpus sencer per a determinar les probabilitats de les futures paraules, sinó que sol dividir-lo en dues parts: una part d'entrenament i una part de validació.

Aquesta última és utilitzada com indica el seu nom per contrastar quan bé s'ajusten les probabilitats obtingudes, però, també pot ser útil per permetre'ns reajustar les probabilitats dels n-grames que no haguem pogut observar durant l'entrenament. És en aquest cas que fem servir l'anomenat estimador *held out*.

Consideris $C_1(w_1, ..., w_n)$ el nombre de vegades que apareix l'n-grama $(w_1, ..., w_n)$ en el nostre conjunt de dades d'entrenament, i sigui $C_2(w_1, ..., w_n)$ el nombre de vegades que apareix el mateix n-grama però en el conjunt de dades de validació. Definim aleshores W_r com el conjunt de n-grames que apareixen exactament r cops en el nostre conjunt de dades d'entrenament, és a dir:

$$W_r = \{(w_1, ..., w_n) \,|\, C_1(w_1, ..., w_n) = r\}$$

I definim T_r com :

$$T_r = \sum_{w \in W_r} C_2(w)$$

És a dir, T_r és el nombre de vegades que apareixen en el conjunt de dades de validació els n-grames que han aparegut r cops en el conjunt de dades d'entrenament.

Finalment, si anomenem N_r al nombre d'elements de C_r, definim l'estimador *held out* d'un n-grama amb freqüència r (és a dir, $C(w_1, ..., w_n) = r$) com:

$$P_{ho}(w_1, ..., w_n) = \frac{T_r}{N_r N} \tag{4.7}$$

La idea principal darrere d'aquest estimador es fer una estimació a partir de la freqüència entre el text observat i el no observat, escalada pel conjunt total de n-grames possibles. Aquest fet és crucial, ja que representa un canvi de mentalitat total respecte als estimadors anteriors. En lloc de tenir en compte les paraules concretes que formen l'n-grama, el tractem només segons la freqüència amb la que apareix, canviant així les classes d'equivalència en les què ens centrem.

Cross-validation : *Deleted estimator*

El problema de l'anterior estimador radica en el fet que requereix separar el nostre conjunt de dades disponible en un inici en dos subconjunts diferents: un per entrenament i l'altre per inferència mitjançant estimadors tipus *held out*. Seguint aquest procés, verdaderament estem perdent la possible informació que ens donaria estimar la probabilitat tenint en compte els n-grames exactes del text *held out*, i de manera similar, no estem tenint en compte la freqüència de les dades observades per tal de predir n-grames no vistos fins el moment.

Donada aquesta situació, el següent pas a seguir seria intentar trobar un mètode de poder utilitzar tot el nostre conjunt inicial de dades disponibles tant per a entrenament com per a validacions *held out*. És a dir, utilitzarem totes les nostres dades tant per a estimació de probabilitats mitjançant freqüències i per *smoothing* dels resultats obtinguts. Aquest conjunt de mètodes solen rebre el nom de mètodes de *cross-validation* o "validació creuada".

Un dels estimadors dins d'aquesta categoria és l'anomenat *deleted estimator*.

Per tal d'entendre'l, suposem que hem dividit totes les nostres dades disponibles d'entrenament T en dos subconjunts T_0 i T_1, complint $T_1 \cup T_2 = T$ i $T_1 \cap T_2 = \emptyset$. Sota aquestes condicions, definim N_r^a com el nombre de n-grames que apareixen exactament r cops en la a-èssima part del text, i T_r^{ab} el nombre de n-grames amb freqüència r en la part a-èssima que apareixen també en la part b-èssima. Aquestes definicions són molt similars a les de l'estimador *held out*, i pot observar-se que si considerem T_0 com les nostres dades d'entrenament i T_1 com les de validació aleshores, per en un n-grama $(w_1, ...w_r)$ amb freqüència r podem estimar la seva probabilitat com:

$$P_{ho}(w_1, ..., w_n) = \frac{T_r^{01}}{N_r^0 N} \tag{4.8}$$

El *deleted estimator* amplia aquesta definició de manera que puguem tenir en compte ambdues parts:

$$P_{del}(w_1, ..., w_n) = \frac{T_r^{01} + T_r^{10}}{N(N_r^0 + N_r^1)} \tag{4.9}$$

És possible generalitzar aquest mètode per dividir el conjunt de dades en K parts, i anar alternant cada una d'aquestes com a dades *held out* i la resta com al conjunt d'entrenament. És a dir, sota les condicions:

$$T = \bigcup_{i=1}^{K} T_i$$
$$T_i \cap T_j = \emptyset \ , \forall i \neq j$$

Tenim que el deleted estimator generalitzat passa a ser:

$$P_{del}(w_1, ..., w_n) = \frac{1}{K} \sum_{i=1}^{K} \frac{T_r^{ij} T_r^{ji}}{N(N_r^i + N_r^j)} \tag{4.10}$$

On j no representa verdaderament la part j-èssima, sinó que és la notació escollida usada per representar la part de les dades d'entrenament no contingudes en T_i.

Intuïtivament, calculem el *deleted estimator* per a cada i-èssima part i en fem una mitjana. La gràcia darrere d'aquest estimador és que podem tenir una visió més general de la informació que ja ens havia introduït l'estimador *held out* fent servir la part "reservada" del text per a predir comportaments no observats de manera rotativa.

Estimador Good-Turing

La naturalesa "rotativa" del *deleted estimator* permet evitar l'*overfitting* dels nostres models als corpus d'entrenament, donant lloc així a models que s'adapten millor a predir les possibilitats d'n-grames no vists comportant això un major rendiment a nivell general.

No obstant, aquest mètode presenta problemes considerables a l'hora del preprocessament de les dades d'entrenament. Per començar, s'introdueix el problema de trobar la millor manera de dividir el corpus per tal d'obtenir subconjunts que realment siguin representatius i no causin segregació de les dades, és a dir, que concentrin grans quantitats de n-grames poc habituals en un mateix subconjunt, donant lloc a freqüències inflades poc representatives de la realitat.

Donada aquesta situació és comprensible estudiar altres estimadors que no requereixin del fraccionament de les dades d'entrenament. És aquí on entra el treball realitzat per Good [14] que, basant-se en la feina de Turing, ens permet introduir un mètode per estimar la probabilitat dels n-grames sota l'assumpció que segueixen una distribució binomial. Tot i que evidentment, la presència de paraules en un text no segueix una distribució binomial, aquest mètode s'ajusta bé a corpus amb vocabularis grans [19, p.212].

L'estimador de Good-Turing és en aparença senzill. Si $(w_1, ..., w_n)$ és un n-grama i N la quantitat total de n-grames del corpus, la probabilitat donada per l'estimador és:

$$P_{GT}(w_1, ..., w_n) = \frac{r^*}{N} \tag{4.11}$$

On r^* és l'anomenada "freqüència ajustada" del nostre n-grama, que ve donada per l'expressió:

$$r^*(w_1, ..., w_n) = (r+1)\frac{E(N_{r+1})}{E(N_r)} \tag{4.12}$$

On r representa la freqüència de l'n-grama i N_r, N_{r+1} són variables aleatòries associades al nombre d'n-grames amb freqüència r i $r+1$ respectivament. Quan $r = 0$, es sol prendre el valor $E(N_1)/N$ com a freqüència ajustada.

Aquest resultat és conseqüència del següent teorema, atribuït a Good[10]:

Teorema 4.3. *Siguin $B_1(N; p_1, ..., p_s)$ i $B_2(N; p_1, ..., p_s)$ dues mostres independents marginalment binomials. Aleshores la freqüència esperada en B_2, r^*, dels elements que ocorren r vegades en B_1 ve donada per l'expressió:*

$$r^* = \frac{r+1}{1+(1/N)}\frac{E(N_{r+1}|B(N+1; p_1, ...p_s))}{E(N_r|B(N; p_1, ...p_s)}$$

Quan ens referim a una mostra notada per $B(N; p_1, ..., p_S)$, on els subíndex indiquen l'ordre en que han estat preses, volem dir que prenem N elements que poden ser de S

tipus diferents , on escollir un element d'un tipus determinat i segueix una distribució binomial p_i.

Que les mostres siguin independents marginalment binomials indica que la seva distribució individual és binomial i que, dins de cada mostra, la probabilitat de seleccionar un element d'un tipus no altera la probabilitat de seleccionar un element d'un altre tipus. Tanmateix, quan es consideren ambdues mostres conjuntament, poden comportar-se de manera diferent a com ho farien si fossin dos experiments completament separats.

Tenint en compte aquestes consideracions, passem a demostrar el teorema:

Demostració. Comencem realitzant tres eleccions de manera independent i simultània. La primera consisteix en escollir un dels S tipus disponibles, assumint una probabilitat uniforme $1/S$ per a cada un d'ells. Les altres dues eleccions generen les mostres $B_1(N; p_1, ..., p_S)$ i $B_2(N; p_1, ..., p_S)$.

Sigui $\widetilde{\mu}$ la variable aleatòria de l'índex resultant en la primera elecció. Siguin $\widetilde{R}_{\widetilde{\mu}1}$ i $\widetilde{R}_{\widetilde{\mu}2}$ la freqüència dels elements del tipus indicat per $\widetilde{\mu}$ en la primera i segona mostra respectivament.

Aleshores podem reformular el teorema com :

$$r^* \equiv E_{012}(\widetilde{R}_{\widetilde{\mu}2} | \widetilde{R}_{\widetilde{\mu}1} = r)$$

Per tal de completar la prova, cal tenir en compte els lemes següents:

Lema 4.3. $E_{012}(\widetilde{R}_{\widetilde{\mu}2} | \widetilde{R}_{\widetilde{\mu}1} = r) = NE_{01}(p_{\widetilde{\mu}} | \widetilde{R}_{\widetilde{\mu}1} = r)$

Lema 4.4. *Consideris s_v, el tipus d'element amb índex v, i la seva respectiva probabilitat p_v Aleshores:*

$$E_{01}(s_v = s_{\widetilde{\mu}} | \widetilde{R}_{\widetilde{\mu}1} = r) = \frac{p_v^r (1 - p_v)^{N-r}}{\sum_{\lambda=1}^{S} p_\lambda^r (1 - p_\lambda)^{N-r}}$$

Lema 4.5. *Consideris E_N l'esperança de qualsevol mostra d'N elements presa (ja sigui B_1 o B_2), N_r el nombre de tipus diferents d'elements que tenen freqüència r. Aleshores:*

$$E_N(N_r) = \binom{N}{r} \sum_{\lambda=1}^{S} p_\lambda^r (1 - p_\lambda)^{N-r}$$

A partir d'aquests, es compleix:

$$\begin{aligned}
r^* &= E_{012}(\widetilde{R}_{\tilde{\mu}2}|\widetilde{R}_{\tilde{\mu}1} = r) \\
&= NE_{01}(p_{\tilde{\mu}}|\widetilde{R}_{\tilde{\mu}1} = r) \text{ [Lema 1]} \\
&= N\sum_{v=1}^{S} p_v E_{01}(s_v = s_{\tilde{\mu}}|\widetilde{R}_{\tilde{\mu}1} = r) \\
&= N\sum_{v=1}^{S} p_v \frac{p_v^r(1-p_v)^{N-r}}{\sum_{\lambda=1}^{S} p_\lambda^r(1-p_\lambda)^{N-r}} \text{ [Lema 2]} \\
&= N\frac{\sum_{v=1}^{S} p_v^{r+1}(1-p_v)^{N-r}}{\sum_{\lambda=1}^{S} p_\lambda^r(1-p_\lambda)^{N-r}} \\
&= N\frac{E_{N+1}(N_r+1)/\binom{N+1}{r+1}}{E_N(N_r)/\binom{N}{r}} \text{ [Lema 3]} \\
&= N\frac{\binom{N}{r}E_{N+1}(N_r+1)}{\binom{N+1}{r+1}E_N(N_r)} \\
&= \frac{r+1}{1/(N+1)}\frac{E_{N+1}(N_{r+1})}{E_N(N_r)}
\end{aligned}$$

Provant així el resultat que es desitjava. □

Passem ara a provar els tres lemes emprats:

Lema 1. Per $i = 1, ..., N$ i per $v \in 1, ..., S$, on S és el nombre de tipus diferents, definim:

$$\tilde{x}_{vi} = \begin{cases} 1 & \text{si l'element } i \text{ es del tipus } v, \\ 0 & \text{altrament.} \end{cases}$$

Aleshores:

$$\tilde{R}_{\tilde{\mu}2} = \sum_{i=1}^{N} \tilde{x}_{\tilde{\mu}i}$$

I per tant:

$$\begin{aligned}
E_{012}(\tilde{R}_{\tilde{\mu}_2}|\tilde{R}_{\tilde{\mu}_1} = r) &= \sum_{i=1}^{N} E_{012}(\tilde{x}_{\tilde{\mu}_i} \mid \tilde{R}_{\tilde{\mu}_1} = r) \\
&= \sum_{i=1}^{N} E_{012}(p_{\tilde{\mu}} \mid \tilde{R}_{\tilde{\mu}_1} = r) \\
&= NE_{01}(p_{\tilde{\mu}} \mid \tilde{R}_{\tilde{\mu}_1} = r)
\end{aligned}$$

La primera igualtat l'obtenim per definició dels $\tilde{x}_{vi}$ i la linealitat de l'esperança. La segona és conseqüència del fet que $\tilde{x}_{vi}$ pren valor 1 amb probabilitat $p_{\tilde{\mu}}$ (mantenim la

condicionalitat de la freqüència presa en la primera mostra perquè recordem que les mostres només son marginalment independents, no conjuntament).

L'última ve donada perquè el sumatori ja no depèn de l'index i, i $p_{\tilde{\mu}}$ només ve determinada per l'elecció del tipus d'element (elecció 0, i per tant, subíndex 0) i la freqüència presa de la primera mostra (elecció 1, subíndex 1), de manera que podem obviar allò obtingut en la segona mostra. □

Lema 2. Sigui $f(v)$ l'event en el qual $s_v = s_{\tilde{\mu}}$, és a dir, en el qual l'índex de l'element coincideix amb l'índex escollit aleatòriament. Sigui $e(r)$ l'event que $s_{\tilde{\mu}}$ aparegui r cops i sigui e(v,r) l'event en el qual $s_v = s_{\tilde{\mu}}$ i s_v ocorre r vegades.

A partir d'aquestes definicions, es segueix que $e(v,r) = f(v) \cap e(r)$ i a més :

$$\bigcup_{\lambda=1}^{S} e(\lambda, r) = \bigcup_{\lambda=1}^{S} (f(\lambda) \cap e(r)) = e(r)$$

on els $\{e(\lambda, r)\}_{\lambda=1,\dots,S}$ són mútuament exclusius.

Per tant :

$$E_{01}(s_v = s_{\tilde{\mu}} \mid R_{\tilde{\mu}1} = r) = P(v = \tilde{\mu} \mid R_{\tilde{\mu}1} = r) = P(f(v) \mid e(r))$$

$$= \frac{P(f(v) \cap e(r))}{P(e(r))} = \frac{P(e(v,r))}{P(e(r))}$$

$$= \frac{P(e(v,r))}{\sum_{\lambda=1}^{S} P(e(\lambda, r))}$$

On la primera igualtat ve donada pel fet que el tipus d'element s_v només coincidirà amb l'escollit en la primera elecció si l'index v coincideix amb l'índex escollit $\tilde{\mu}$, i per tant podem tractar $s_v = s_{\tilde{\mu}}$ com una variable aleatòria indicador que pren valor 1 en la situació anterior.

Ara, com $\tilde{\mu}$ és escollit amb probabilitat uniforme $1/S$ i r_v segueix una distribució binomial amb probabilitat p_v, tenim :

$$P(e(v,r)) = \frac{1}{S}\binom{N}{r} p_v^r (1 - p_v)^{N-r}$$

Del que finalment s'obté:

$$E_{01}(s_v = s_{\tilde{\mu}} \mid R_{\tilde{\mu}1} = r) = \frac{\frac{1}{S}\binom{N}{r} p_v^r (1 - p_v)^{N-r}}{\sum_{\lambda=1}^{s} \frac{1}{S}\binom{N}{r} p_\lambda^r (1 - p_\lambda)^{N-r}}$$

$$= \frac{p_v^r (1 - p_v)^{N-r}}{\sum_{\lambda=1}^{s} p_\lambda^r (1 - p_\lambda)^{N-r}}$$

□

Lema 3. Per a qualsevol r definim:

$$N_r = \sum_{v=1}^{S} \delta(r, r_v)$$

On $\delta(r, f)$ és la funció delta, és a dir:

$$\delta(r,f) = \begin{cases} 1 & r = f \\ 0 & r \neq f \end{cases}$$

Aleshores:

$$\begin{aligned} E(N_r|B(N;p_1\ldots,p_S)) &= \sum_{v=1}^{S} \delta(r,r_v) P(r_v \mid B(N;p_1\ldots,p_S)) \\ &= \sum_{v=1}^{S} P(r \mid B(N;p_1\ldots,p_S)) \\ &= \sum_{v=1}^{S} \binom{N}{r} p_v^r (1-p_v)^{N-r} \\ &= \binom{N}{r} \sum_{v=1}^{S} p_v^r (1-p_v)^{N-r} \end{aligned}$$

On la primera igualtat ve donada per la linearitat de l'esperança i la resta es segueixen de la definició de la funció delta i de que la mostra segueix una distribució binomial per a cada tipus d'element v.

□

Concloent així la demostració del teorema. Ara bé, cal entendre que aquest no és emprat directament, sinó que prenem la següent aproximació per a valors "grans" de N :

$$r^* = \frac{r+1}{1+(1/N)} \frac{E(N_{r+1}|B(N+1;p_1,...p_s))}{E(N_r|B(N;p_1,...p_s)} \approx (r+1)\frac{E(N_r+1 \mid B(N))}{E(N_r|B(N))} \tag{4.13}$$

Aquesta aproximació té un error relatiu de $1/N$, i podem considerar bones les aproximacions per a $N \geq 10^7$ [10].

A més a més, en la pràctica es solen substituir les esperances $E(N_{r+1})$ i $E(N_r)$,pels valors trobats empíricament N_r i N_{r+1} quan estem treballant amb corpus grans.

No obstant, aquest procés no es pot aplicar de manera uniforme, ja que es presenta un gran inconvenient : seguint aquesta pràctica, els n-grames més freqüents tindran probabilitat estimada 0, ja que per en aquests $N_{r+1} = 0$, en altres paraules, cap n-grama pot aparèixer més que aquells que tenen freqüència màxima.

Per tal de solucionar aquest problema, es solen utilitzar dos mètodes. El primer consisteix en només usar l'estimador de Good-Turing per a valors complint $r < k$ on k és una constant fixada. L'altra opció consisteix a trobar una funció $S(r)$ que s'ajusti a les observacions realitzades dels N_r i usar els valors d'aquesta per a substituir les esperances, i distribuir uniformement les probabilitats per als n-grames no observats.

4.2 Aplicacions i combinació de models

La majoria dels estimadors explicats en aquesta secció no s'utilitzen de manera individual, sinó que solen ser usats en conjunt per tal de determinar probabilitats condicionades o

desconegudes. Una de les situacions més comunes és fer servir l'estimador de Good-Turing amb el de màxima versemblança per tal de calcular probabilitats condicionades. Per exemple, suposem que volem calcular la possibilitat que aparegui la paraula *she*, després de *person*. És a dir, volem calcular la probabilitat $P(she \mid person)$. Tenint en compte el teorema de Bayes, és equivalent a calcular:

$$P(she \mid person) = \frac{P(person, she)}{P(she)} = \frac{P_{GT}(person, she)}{P_{MLE}(she)}$$
$$= \frac{r^*(person, she)/N_b}{C(she)/N} = \frac{r^*(person, she)}{C(she)} \cdot \frac{N_b}{N}$$

On apliquem al numerador la probabilitat donada per l'estimador de Good-Turing i al denominador la probabilitat de l'estimador de màxima versemblança (en cas que aquesta paraula hagi aparegut, altrament podríem fer ús d'un altre com per exemple l'estimador de Laplace). Simplificant el darrer quocient ($N_b = N - 1$ si no hem afegit cap paraula de "padding", i per tant $\frac{N}{N-1} \approx 1$ per corpus grans), obtenim el quocient entre la freqüència ajustada per Good-Turing i el recompte total de l'anterior element. Aleshores en la pràctica, simplement s'usaria l'expressió:

$$P(she \mid person) = \frac{r^*(person, she)}{C(she)}$$

Per exemple, fixant-nos en els valors obtinguts estudiant el corpus d'Austen (que inclou 6 novel·les de Jane Austen [19, p.214]), obtenim per al bigrama buscat el següent resultat:

$$P(she \mid person) = \frac{r^*(person, she)}{C(she)} = \frac{(r+1)N_3}{C(she)N_2} = \frac{3 \cdot 10531}{223 \cdot 24531} = 0.0055$$

Per altra banda, un dels mètodes més utilitzats en la pràctica per abordar les limitacions dels models basats en un únic tipus d'n-grames és combinar-los amb altres models. Per exemple, és habitual complementar un model de trigrames amb models de bigrames i unigrames mitjançant tècniques d'interpolació, ajustant així les probabilitats assignades a cada element per obtenir estimacions més robustes.

La forma més senzilla és la interpolació lineal, habitualment usada amb un model basat en trigrames, però fàcilment extensible a qualsevol ordre superior :

$$P(w_n \mid w_{n-2}, w_{n-1}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n|w_{n-1}) + \lambda_3 P_3(w_n|w_{n-1}, w_{n-2}) \qquad (4.14)$$

On els λ_i compleixen que $0 < \lambda_i < 1$ i $\sum_i \lambda_i = 1$.

Aquests valors es poden assignar manualment o ser calculats mitjançant algoritmes com es veurà en seccions posteriors.

Conclusions dels n-grames

En aquesta secció hem explorat els conceptes fonamentals dels models de llenguatge basats en n-grames, destacant el seu paper en la modelització estadística del llenguatge. Aquests

models capturen les relacions probabilístiques entre seqüències de paraules, oferint un marc pràctic i computacionalment eficient per aproximar la complexitat del llenguatge natural. Tot i això, no estan exempts de limitacions, com ara la sensibilitat a la falta de dades i la incapacitat de captar dependències a llarg abast en el text.

Per abordar aquestes limitacions, s'utilitzen tècniques com el suavitzat i la interpolació, que permeten combinar diversos models d'n-grames per equilibrar la precisió i la generalització. Malgrat que aquestes metodologies mitiguen alguns dels reptes inherents, posen de manifest la necessitat de mètodes més sofisticats que puguin aprofitar estructures lingüístiques més riques.

Aquesta anàlisi ens prepara per explorar els avenços moderns en la modelització del llenguatge, com ara les arquitectures neuronals basades en el Transformer, que superen moltes de les limitacions discutides. Aquests nous models representen una evolució natural dels principis establerts pels enfocaments basats en n-grames, combinant el rigor estadístic amb la capacitat de modelar dependències a llarg abast i patrons complexos en el text.

4.3 Cadenes de Màrkov per a la modelització del llenguatge

Tot i que els models basats en n-grames ofereixen una manera pràctica i computacionalment eficient d'aproximar el llenguatge natural, aquests poden considerar-se un cas específic d'un marc més general: els models de cadenes de Màrkov. Les cadenes de Màrkov generalitzen la idea subjacent als n-grames on considerem que una paraula depèn d'una part limitada del text que la precedeix; això és una extensió natural de l'assumpció de Màrkov, que estableix que la probabilitat d'un esdeveniment futur només depèn de l'estat present i no del passat complet.

De manera formal, definim una cadena de Màrkov de la següent manera [6, p.88]:

Definició 4.6. *Sigui S un conjunt finit o numerable, i (Ω, F, P) un espai de probabilitats. Aleshores es defineix una cadena de Màrkov amb valors sobre S com una successió de variables aleatòries X_n sobre l'espai de probabilitats, $n \in N$, si $\forall n \in \mathbb{N}$ i $\forall s \in S$ es compleix :*

$$P(X_{n+1} = s \mid X_0, \ldots, X_n) = P(X_{n+1} = s \mid X_n) \tag{4.15}$$

En altres paraules, una cadena de Màrkov és una successió de variables aleatòries que compleix la propietat o assumpció de Màrkov. Sovint parlem dels estats i el conjunt d'estats d'una cadena de Màrkov. La definició d'aquests conceptes és senzilla:

Definició 4.7. *El conjunt d'estats d'una cadena de Màrkov és el conjunt finit o numerable S sobre el qual la definim. A cada element $s \in S$ se l'anomena estat.*

En seccions anteriors, s'ha detallat com sovint en la pràctica es sol considerar el llenguatge com un procés estocàstic estacionari. De forma similar, a l'hora de modelar llenguatges mitjançant cadenes de Màrkov, fem servir un tipus particular anomenat cadenes de Màrkov homogènies:

Definició 4.8. *Sigui X_n, $n \in \mathbb{N}$ una cadena de Màrkov amb valors sobre S. Diem que X_n és homogènia si $\forall n \in \mathbb{N}$ i $\forall i, j \in S$ es compleix:*

$$P(X_{n+1} = j \mid X_n = i) = P(X_1 = j \mid X_0 = i) \tag{4.16}$$

Per simplificar la notació, reescrivim $P(X_1 = j \mid X_0 = i)$ com $p(j \mid i)$ i l'anomenem la probabilitat de transició de l'estat i a l'estat j. Podem doncs representar una cadena de Màrkov mitjançant l'anomenada matriu de transició:

Definició 4.9. *Sigui X_n una cadena de Màrkov sobre S. Definim la matriu de transició P de X_n com :*

$$P = (p(j \mid i))_{\forall i,j \in S} \tag{4.17}$$

En altres paraules, és la matriu on l'element de la i-èssima columna i j-èssima fila és la probabilitat de transició de l'estat i a l'estat j.

En general, per tota matriu de transició P tenim:

- Al tractar-se de cada element de la matriu una probabilitat : $p_{ji} \geq 0$
- Al ser cada columna el conjunt de totes les transicions d'un estat i a qualsevol estat j: $\sum_{j \in S} p_{ji} = 1$

A les matrius que compleixen aquestes condicions se les anomena matrius estocàstiques.

Observem però que aquesta matriu no està completa, sinó que seria necessari a més incloure un vector de probabilitats inicials que donés la probabilitat amb la què X_0 pren un valor determinat del conjunt S. Generalment notem aquest vector per $\Pi = (\pi_i)_{i \in S}$ on $\pi_i = P(X_0 = i)$.

A la pràctica però, aquest vector es sol ometre, i es substitueix mitjançant l'addició d'un estat inicial extra s_0 al conjunt S, de manera que la matriu P disposi d'una columna i una fila addicionals.

A partir d'això, hom pot veure que una cadena de Màrkov homogènia queda doncs completament definida per la seva matriu de transició.

A més a més, si el conjunt d'estats no és excessivament gran, es pot arribar a representar la cadena gràficament mitjançant un diagrama d'estats.

Per tal d'exemplificar aquest concepte, considerem la següent situació :

Estem intentant identificar un llenguatge desconegut trobat en un document que es troba incomplet i erosionat pel pas del temps, de manera que només unes poques lletres són distingibles. A més, els pocs caràcters que podem distingir estan agrupats en síl·labes de com a màxim tres lletres, tot i que la majoria estan formades per dues.

Després d'analitzar el document detingudament, determinem que només apareixen les lletres (t, h, p, i, a, e).

Donada aquesta situació, decidim fer ús d'una cadena de Màrkov per representar la situació, ja que disposem d'un conjunt finit de possibles estats (lletres), i no tenim suficient context per poder inferir una dependència entre lletres distants entre si (síl·labes curtes).

En conseqüència, procedim a calcular les seves probabilitats condicionades tenint únicament en compte la lletra anterior. Finalment, decidim a més que considerarem la lletra t com el nostre punt inicial, ja que és la primera lletra que es pot distingir en el document seguint l'ordre habitual de lectura.

Donat que només tenim sis lletres possibles, decidim representar els resultats obtinguts mitjançant el diagrama d'estats representat en la figura 3.

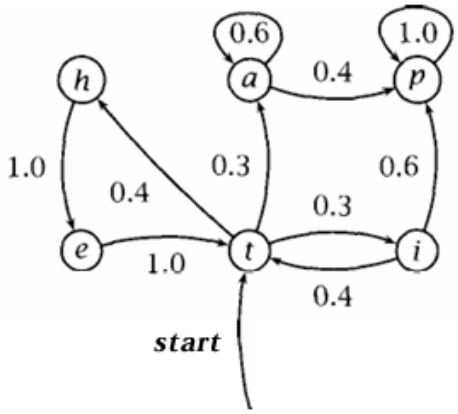

Figura 3: Diagrama d'estats, extret de [19]

En aquest diagrama, cada node representa un estat, i cada fletxa una de les seves possibles transicions on el nombre que les acompanya és la probabilitat que es doni cadascuna d'aquestes. De fet, és senzill poder recuperar la matriu de transicions P d'aquesta cadena a partir del diagrama :

$$P = \begin{bmatrix} 0.6 & 0 & 0 & 0 & 0.3 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.4 & 0 \\ 0 & 0 & 0 & 0 & 0.3 & 0 \\ 0.4 & 0 & 0 & 0.4 & 1.0 & 0 \\ 0 & 1.0 & 0 & 0.6 & 0 & 0 \end{bmatrix}$$

On cal tenir en compte que s'ha decidit ordenar les lletres alfabèticament. Per facilitar la interpretació, a continuació es detalla amb una taula de doble entrada:

	a	e	h	i	p	t
a	0.6	0	0	0	0	0.3
e	0	0	1.0	0	0	0
h	0	0	0	0	0	0.4
i	0	0	0	0	0	0.3
p	0.4	0	0	0.6	1.0	0
t	0	1.0	0	0.4	0	0

I el nostre vector de probabilitats inicials és $\Pi = (0, 0, 0, 0, 0, 1)$, ja que sabem que la primera lletra observada és la t.

La matriu de transició ens ofereix una manera clara i concisa de determinar les probabilitats que pot prendre una seqüència d'estats determinada, i en el cas de les cadenes de Màrkov homogènies, determina completament el comportament d'aquesta.

Com a conseqüència de la propietat de Màrkov, el càlcul de les probabilitats d'una seqüència determinada d'elements es veu simplificada:

$$\begin{aligned} P(X_0, \ldots, X_N) &= P(X_0)P(X_2 \mid X_1)P(X_3 \mid X_2, X_1) \ldots P(X_N \mid X_{N-1}, \ldots, X_1) \\ &= P(X_0)P(X_2 \mid X_0)P(X_3 \mid X_2) \ldots, P(X_N \mid X_{N-1}) \\ &= \pi_{X_0} \cdot \prod_{t=1}^{N} P(X_t \mid X_{t-1}) \end{aligned}$$

Així doncs, remetent-nos al nostre exemple, calcular la probabilitat de que aparegui la seqüència (t, h, e) com a la primera síl·laba del text és:

$$\begin{aligned} P(X_0 = t, X_1 = h, X_2 = e) &= (X_0 = t) \cdot P(X_1 = h \mid X_0 = t) \cdot P(X_2 = e \mid X_1 = h) \\ &= \pi_6 \cdot p_{36} \cdot p_{23} = 1 \cdot 0.4 \cdot 1.0 = 0.4 \end{aligned}$$

Sovint en la pràctica es solen aplicar cadenes de Màrkov per modelar processos que de manera natural semblen no seguir un comportament similar. Així doncs, una de les qüestions més importants es si podem representar el nostre objecte d'estudi mitjançant una cadena de Màrkov, i no que ho fem de manera instintiva.

Prenguis per exemple el cas de la modelització per n-grames quan $n \geq 2$ (en el cas dels unigrames, simplement li assignem a cada paraula una probabilitat). En el cas dels bigrames és fàcil veure com podem elaborar una cadena de Màrkov per representar el llenguatge:

- Prenem com a S el conjunt de totes les paraules diferents del nostre corpus ordenades en ordre alfabètic
- Determinem el vector de probabilitats inicials Π, assignant a cada paraula una probabilitat mitjançant un estimador de la nostra selecció, per exemple, l'estimador de màxima versemblança
- Elaborem P fent ús de les probabilitats condicionades calculades mitjançant la combinació de dos estimadors, com el de Good-Turing i el de versemblança, tal i com s'ha explicat en seccions anteriors.

En general, podem modelar qualsevol model per n-grames com una cadena de Màrkov on cada estat ve representat per un $n-1$ grama:

- Prenem com a conjunt d'estats S aquell format per tots els $n-1$ grames possibles.
- Determinem el vector de probabilitats inicial Π assignant-li les probabilitats inicials dels $n-1$ grames donades per la cadena de Màrkov anterior.
- Elaborem P fent ús de les probabilitats calculades mitjançant l'ús d'estimadors pertinents, col·locant aquestes en la posició ocupada pels $n-1$ grames solapats entre si. És a dir, la probabilitat de l'n-grama $(w_1, \ldots, w_n)$ estaria en la posició de la matriu que indiqués la transició entre els n-1 grames $(w_1, \ldots, w_{n-1})$ i $(w_2, \ldots, w_n)$. Per exemple, si estem treballant amb trigrames, la probabilitat del trigrama *the air was* seria la probabilitat de transició entre l'estat determinat pel bigrama *the air* i el bigrama *air was*.

És a dir, l'ús de cadenes de Màrkov per a la modelització del llenguatge ja estava "implícit" en la modelització per n-grames.

De fet, les cadenes de Màrkov solen ser usades en general per modelar llenguatges per la seva capacitat de capturar les relacions seqüencials entre elements, com poden ser síl·labes o paraules. A més, la propietat de Màrkov permet capturar en certa part el comportament reglat de les normes lingüístiques.

La naturalesa del llenguatge però implica que hem de fer una petita consideració abans de fer ús de les cadenes de Màrkov. En els exemples anteriors, cada un dels estats possibles de la cadena eren deterministes, és a dir, cada estat implicava una observació determinada.

A la pràctica, s'utilitzen els anomenats *Hidden Màrkov Models* o models ocults de Màrkov [19, p.320], on cada estat determina quina probabilitat tindrà cada una de les possibles observacions del model. Aquests models defineixen dues probabilitats diferents: una probabilitat entre estat i estat (la mateixa que en el model de Màrkov bàsic observat anteriorment) i la probabilitat entre estat i observació (coneguda com *probabilitat d'emissió*).

Definició 4.10. *Un model ocult de Màrkov (Hidden Màrkov Model) és una 5-tupla* (S, K, Π, A, B) *on :*

- $S = (s_1, \ldots, s_n)$ *és el conjunt d'estats.*
- $K = (k_1, \ldots, k_t)$ *és el conjunt d'observacions.*
- $\Pi = (\pi_1, \ldots, \pi_n)$ *és el conjunt de probabilitats inicials de cada estat*
- $A = (a_{ij})_{i,j \in S}$ *és la matriu de transició, on l'element de la* i*-èssima fila i la* j*-èssima columna és la probabilitat de transició de l'estat* j *a l'estat* i*,* $p(i \mid j)$*.*
- $B = \{b_{ijk}\}\, i, j \in S,\, k \in K$ *és el conjunt de probabilitats d'emissió de cada observació. És a dir,* $b_{ijk} = P(O_t = k \mid X_{t+1} = j, X_t = i, \mu)$*.*

A cada model ocult de Màrkov se li associa a més:

- *Una seqüència d'estats* $X = (X_1, \ldots, X_N)$ *on cada* $X_i : S \to \{1, \ldots, n\}$ *és una variable aleatòria*
- *Una seqüència d'observacions* $O = (o_1, \ldots, o_N)$ *on* $o_i \in K$

En algunes definicions de models ocults de Màrkov, la probabilitat d'emissó només depen de l'estat anterior. A l'hora però d'aplicar aquests en l'àmbit lingüístic, es determina la probabilitat d'emissió a partir de la transició entre estats donats, ja resulta més important el canvi entre paraules que no només la paraula anterior. Donada la naturalesa d'aquest treball, s'ha optat per donar la definició basada en la transició d'estats.

Un exemple de model ocult de Màrkov aplicat al llenguatge el podem trobar en la combinació de diversos models per n-grames. Recordem que vam introduir la interpolació lineal entre unigrames, bigrames i trigrames com a solució per complementar l'escassetat de dades en els trigrames de la següent forma :

$$P_{li}(w_n \mid w_{n-1}, w_{n-2}) = \lambda_1 P_1(w_n) + \lambda_2 P_2(w_n \mid w_{n-1}) + \lambda_3 P_3(w_n \mid w_{n-1}, w_{n-2})$$

On tenim a més la restricció $\sum_{i=1}^{3} \lambda_i = 1$. Podem representar aquesta interpolació com un model ocult de Màrkov on cada estat representa l'opció d'escollir un unigrama, un bigrama o un trigrama. Més en concret, per a cada parella inicial de paraules (w_a, w_b) construïm quatre estats :

1. Per a cada parella bàsica de paraules.
2. Per a la probabilitat de la tercera paraula escollida per unigrames.
3. Per a la probabilitat de la tercera paraula escollida per bigrames.
4. Per a la probabilitat de la tercera paraula escollida per trigrames.

Representat en un diagrama, obtindríem un resultat similar a la figura 4. Observis que cada transició entre estats ve denotada per $o : p$ on p indica la probabilitat de la transició i o indica l'output resultant després d'aquesta transició.

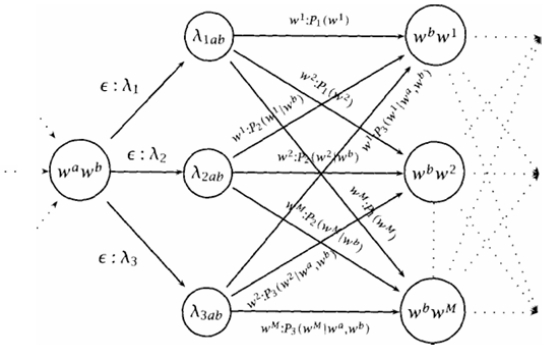

Figura 4: Diagrama del HMM amb interpolació lineal dels trigrames, extret de [19]

Cal destacar també l'aparició dels estats buits, λ_{iab} amb transició $\epsilon : \lambda_i$, que només indiquen quin dels 3 models d'n-grames s'utilitza per a determinar la probabilitat de la paraula següent, però sense donar cap output (per això la notació ϵ).

Després de cada un d'aquests estats, ens traslladem a cada una de les següents M possibles paraules amb la probabilitat del model seleccionat, i així successivament fins a finalitzar.

L'avantatge principal d'usar aquests models és que existeixen algoritmes per a entrenar-los i trobar els millors paràmetres λ_{iab}.

Abans però d'entrar en detall sobre aquest, cal realitzar-nos tres preguntes principals sobre els models ocults de Màrkov, que ens ajudaran a posar context al problema i entendre els passos realitzats :

1. Donat un model $\mu = (A, B, \Pi)$, com calculem la probabilitat d'una observació determinada O, és a dir, com es troba $P(O \mid \mu)$?
2. Donada una observació O i un model μ com trobem una seqüència d'estats $(X_1, \ldots, X_N)$ que millor la descrigui?
3. Donada una observació O i una família de models obtinguda variant els paràmetres $\mu = (A, B, \Pi)$, com obtenim el model que millor s'ajusta a les dades observades?

És fonamental respondre aquestes tres qüestions per tal d'entendre l'aplicació i funcionament dels models ocults de Màrkov.

Càlcul de $P(O \mid \mu)$

Comencem responent a la primera pregunta. El procés de calcular $P(O \mid \mu)$ és anomenat el procés de descodificació, i de forma "ingènua", podem entendre'l de la següent manera:

Considerant una seqüència d'observacions $O = (o_1, \ldots, o_N)$ i una seqüència d'estats $(X_1, \ldots, X_{N+1})$, es compleix:

$$P(O \mid X, \mu) = \prod_{i=1}^{N} P(o_i \mid X_i, X_{i+1}, \mu) = b_{X_1 X_2 o_1} \cdot b_{X_2 X_3 o_2} \cdot \ldots \cdot b_{X_N X_{N+1} o_N} \tag{4.18}$$

Per altra banda :

$$P(X \mid \mu) = \pi_{X_1} \prod_{i=1}^{N} a_{X_i X_{i+1}} \tag{4.19}$$

Pel teorema de Bayes es compleix:

$$\begin{aligned} P(A, B \mid C) = \frac{P(A, B, C)}{P(C)} &= \frac{P(A \mid B, C)\, P(B \cap C)}{P(C)} \\ &= \frac{P(A \mid B, C)\, P(B \mid C)\, P(C)}{P(C)} \\ &= P(A \mid B, C)\, P(B \mid C) \end{aligned} \tag{4.20}$$

Motiu pel qual arribem a:

$$P(O, X \mid \mu) = P(O \mid X, \mu)\, P(X \mid \mu) \tag{4.21}$$

Finalment:

$$\begin{aligned} P(O \mid \mu) &= \sum_{X} P(O, X \mid \mu) \\ &= \sum_{X_1, \ldots, X_{N+1}} \pi_{X_1} \prod_{i=1}^{N} a_{X_i X_{i+1}} b_{X_i X_{i+1} o_i} \end{aligned} \tag{4.22}$$

Observis que en cada pas de la nostra seqüència realitzem el producte corresponent a la probabilitat de transició entre estats i la probabilitat d'emissió de l'observació associada en aquella transició, el que resulta en un total de N productes. Donat que a més després hem de realitzar el producte entre tots els resultats anteriors,hem d'afegir $N-1$ productes, acumulant-se $N + N - 1 = 2N - 1$ operacions fins el moment. Si en aquests li afegim el producte amb la probabilitat inicial de la seqüència, aleshores tenim un total de $2N$ productes per seqüència. Finalment, al ser possibles n^{N+1} seqüències, el total de productes requerits per a calcular qualsevol observació donat un model determinat és $2N \cdot n^{N+1}$.

Aquesta aproximació no resulta òptima, motiu pel qual en la pràctica es solen utilitzar algoritmes de programació dinàmica per calcular-los. De manera resumida, en la programació dinàmica l'objectiu és descompondre un problema en "subproblemes" previs,

fins arribar a casos inicials que poden ser calculats directament. A partir d'aquests casos inicials, es calculen la probabilitat dels altres.

Aplicat al cas dels models ocults de Màrkov, l'objectiu consisteix a anar guardant les probabilitats d'acabar en un cert estat donat un camí d'estats recorregut fins al moment mitjançant una matriu.

Formalment, consisteix a anar computant les següents variables:

$$\alpha_i(t) = P(o_1 o_2 \dots o_{t-1}, X_t = i \mid \mu) \tag{4.23}$$

Que representen la probabilitat d'acabar en l'estat i en el moment t donades les observacions anteriors $(o_1, \dots, o_{t-1})$. El càlcul d'aquestes variables es realitza mitjançant el següent procediment, anomenat *Forward Algorithm* descrit en l'algoritme 1.

Aquest algoritme consisteix a construir una matriu amb n files i N columnes, on les files representen els possibles estats i les columnes la seqüència temporal d'observacions. Per a la primera columna, assignem les probabilitats inicials del model directament. Per a les columnes següents, calculem cada valor com el sumatori de totes les probabilitats de la columna anterior ponderades per la probabilitat de transició entre estats corresponents i la probabilitat d'emissió de l'observació després de la transició. Finalment, la probabilitat total de l'observació es calcula com el sumatori de tots els valors de l'última columna, ja que recordem que tenim una seqüència de $N + 1$ estats.

Algorithm 1: Forward Algorithm per Models ocults de Màrkov

1: **Inici:**

$$\alpha_i(1) = \pi_i, \quad 1 \leq i \leq n$$

2: **Inducció:**

$$\alpha_j(t+1) = \sum_{i=1}^{n} \alpha_i(t) a_{ij} b_{ij,o_t}, \quad 1 \leq t \leq N,\ 1 \leq j \leq n$$

3: **Resultat:**

$$P(O \mid \mu) = \sum_{i=1}^{n} \alpha_i(N+1)$$

Observem que aquest algoritme requereix $2n$ productes per calcular cada element a_{ij} d'una columna posterior a la inicial, ja que per a cada element de la fila anterior realitzem dos productes. Com que hi ha un total de n elements en cada columna, això implica $2n^2$ productes per columna. Donat que el nombre total de columnes és N, el nombre total de productes és $2n^2N$. Així, emmagatzemar els resultats intermedis (els estats anteriors) permet reduir significativament el temps necessari per calcular les probabilitats de les observacions, sempre que es disposi de suficient memòria per guardar les variables requerides.

Anàlogament, existeix el *Backward Algorithm* on les variables es computen en l'ordre invers que en l'anterior. Per això, definim les següents variables:

$$\beta_i(t) = P(o_t, \dots, o_T \mid X_t = i, \mu) \tag{4.24}$$

Que representen la probabilitat de veure la resta de l'observació donat que el model es trobava en l'estat i al moment t. A diferència de les variables α_i definides anteriorment,

en aquest cas ens centrem en la part de l'observació que ens queda per veure i no la ja observada. A més, en aquestes es dóna per conegut l'estat, mentre que en les anteriors s'intentava predir la probabilitat de l'estat.

El procediment per determinar les variables ve descrit en l'algoritme 2. Destacar que en aquest el procediment és a la inversa, doncs comencem en l'última columna de la matriu de resultats intermedis i anem calculant les anteriors fins arribar a la primera, d'on obtindrem el resultat realitzant el sumatori de les variables ponderades per la probabilitat inicial de cada estat corresponent. Notis que inicialitzem l'última columna com a 1, ja que en aquest cas, en l'instant (T+1) "no hi ha res a observar", és a dir, no hi ha cap resultat un cop ha acabat la seqüència que s'estava estudiant.

Algorithm 2: Backward Algorithm per models ocults de Màrkov

1: **Inici:**

$$\beta_i(T+1) = 1, \quad 1 \leq i \leq n$$

2: **Inducció:**

$$\beta_i(t) = \sum_{j=1}^{n} a_{ij} b_{j,o_t} \beta_j(t+1), \quad 1 \leq t \leq N,\ 1 \leq i \leq n$$

3: **Resultat:**

$$P(O \mid \mu) = \sum_{i=1}^{n} \pi_i \beta_i(1)$$

La introducció del mètode *Backward* per calcular la probabilitat d'una observació ve del fet que ambdues poden ser combinades alhora:

$$\begin{aligned} P(O, X_t = i \mid \mu) &= P(o_1, \ldots, o_N, X_t = i \mid \mu) \\ &= P(o_1, \ldots, o_{t-1}, X_t = i, o_t, \ldots, o_N \mid \mu) \\ &= P(o_1, \ldots, o_{t-1}, X_t = i \mid \mu) \cdot P(o_t, \ldots, o_N \mid o_1, \ldots, o_{t-1}, X_t = i, \mu) \\ &= P(o_1, \ldots, o_{t-1}, X_t = i \mid \mu) P(o_t, \ldots, o_N \mid X_t = i, \mu) \\ &= \alpha_i(t)\beta_i(t). \end{aligned} \tag{4.25}$$

On la tercera igualtat és conseqüència del teorema de Bayes i la propietat commutativa i associativa de la intersecció de conjunts ja que :

$$P(A, B \mid C) = \frac{P(A, B, C)}{P(C)} = \frac{P(B, (A, C))}{P(C)} = P(B \mid A, C) \cdot P(A \mid C) \tag{4.26}$$

Finalment :

$$P(O \mid \mu) = \sum_{i=1}^{n} \alpha_i(t)\beta_i(t), \quad 1 \leq t \leq N+1 \tag{4.27}$$

Observis que a partir d'aquesta expressió general s'obtenen els resultats de *Forward* quan $t = N+1$ i *Backward* quan $t = 1$.

Selecció de la millor seqüència d'estats

Continuem responent a la segona qüestió plantejada sobre els models ocults de Màrkov. Podem reformular el problema com el trobar el millor estat i en un moment t de la seqüència que maximitzi $P(X_t = i \mid O, \mu)$ donada una observació $O = (o_1, \ldots, o_N)$ fixada.

Sigui doncs:

$$\gamma_i(t) = P(X_t = i \mid O, \mu) = \frac{P(X_t = i, O \mid \mu)}{P(O \mid \mu)} = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{n} \alpha_j(t)\beta_j(t)} \tag{4.28}$$

Per tant, el millor estat en cada moment t seria:

$$\hat{X}_t = \arg \max_{1 \leq i \leq n} \gamma_i(t), \quad 1 \leq t \leq N+1 \tag{4.29}$$

És a dir, prenem una aproximació *greedy* a l'algoritme, el que no necessàriament ens dóna la seqüència d'estats òptima a nivell general ja que no té en compte els estats anteriors que s'hagin seleccionat i, per tant, obvia la rellevància de les probabilitats de transició en la maximització.

Per tal de trobar la solució òptima a nivell global, en comptes d'intentar considerar el millor estat a cada pas t, intentem trobar la millor subseqüència d'estats sencera $X = (X_1, \ldots, X_{N+1})$:

$$\arg \max_{X} P(X \mid O, \mu) \tag{4.30}$$

Pel teorema de Bayes es compleix:

$$P(X \mid O, \mu) = \frac{P(X, O, \mu)}{P(O, \mu)} = \frac{P(X, O \mid \mu) \cdot P(\mu)}{P(O, \mu)} = \frac{P(X, O \mid \mu)}{P(O \mid \mu)} \tag{4.31}$$

I en conseqüència, donat que el denominador no depèn del X escollit i la nostra observació O està fixada, podem afirmar que :

$$\arg \max_{X} P(X \mid O, \mu) \equiv \arg \max_{X} P(X, O \mid \mu) \tag{4.32}$$

Una manera eficient per poder calcular aquest valor és l'algoritme de Viterbi [19, p.332]. En aquest definim les variables:

$$\delta_j(t) = \max_{X_1, \ldots, X_{t-1}} P(X_1, \ldots, X_{t-1}, o_1, \ldots, o_{t-1}, X_t = j \mid \mu) \tag{4.33}$$

A més, introduïm variables de *backtracking* per determinar la seqüència de nodes recorreguda. Si les anteriors determinaven el valor màxim de la probabilitat, aquestes emmagatzemen el valor de l'argument (en aquest cas, l'estat X_t anterior):

$$\psi_j(t+1) = \arg \max_{1 \leq i \leq N} \delta_i(t) a_{ij} b_{ij o_{t+1}}, \quad 1 \leq j \leq N \tag{4.34}$$

El procediment per determinar aquestes variables és el següent:

Observis que per a reconstruir el camí, hem d'iterar per cada una de les variables $\psi_j(t+1)$, començant per N fins arribar a 1 (*backtracking*).

De manera informal, l'algoritme en cada pas t de la seqüència determina dos valors per a cada possible estat j amb $1 \leq j \leq n$:

Algorithm 3: Algoritme de Viterbi

1: **Inici:**

$$\delta_j(1) = \pi_j, \quad 1 \leq j \leq n$$

2: **Inducció:**

$$\delta_j(t+1) = \max_{1 \leq i \leq n} \delta_i(t) a_{ij} b_{jo_{t+1}}, \quad 1 \leq j \leq n$$

$$\psi_j(t+1) = \arg \max_{1 \leq i \leq n} \delta_i(t) a_{ij} b_{jo_{t+1}}, \quad 1 \leq j \leq n$$

3: **Finalització i *backtracking* per determinar camí:**

$$\hat{X}_{N+1} = \arg \max_{1 \leq i \leq n} \delta_i(N+1)$$

$$\hat{X}_t = \psi_{\hat{X}_{t+1}}(t+1), \quad 1 \leq t \leq N$$

$$P(\hat{X}) = \max_{1 \leq i \leq n} \delta_i(N+1)$$

1. $\delta_j(t+1)$: la probabilitat màxima d'arribar a l'estat j en el moment $t+1$ des de l'estat anterior i en el moment t. Per això, usem les variables calculades en el pas anterior ponderades pel seu producte de la probabilitat de transició i emissió del resultat o_{t+1} que ha de ser observat.

2. $\psi_j(t+1)$: L'argument, és a dir, l'estat j que ens ha donat el valor de la variable anterior.

Desglossant la complexitat computacional d'aquest algoritme, obtenim:

- n operacions inicials per determinar els $\delta_j(1)$
- $2n$ productes per determinar cada $\delta_j(t+1)$, que ascendeix a $2n^2$ productes per a tots els j possibles per un t concret; donant lloc a un total de $2n^2N$ productes en la inducció. Si considerem a més que en cada computació actualitzem la variable $\psi_j(t+1)$, obtenim un total de $4n^2N$ operacions.
- n operacions per trobar el millor últim estat, N operacions per reconstruir la seqüencia de millors estats i n operacions per determinar la probabilitat de la millor seqüència. En total, tenim $2n + N$ operacions per al pas de finalització i *backtracking*.

És a dir, podem veure que l'algoritme de Viterbi requereix d'un total de $4n^2N+3n+N$ operacions per completar-se. Generalment, i com s'ha fet en els algoritmes *Forward* i *Backward* anteriors, només solem tenir en compte el nombre de productes, ja que aquests són els més costosos a nivell computacional, en comparació amb les operacions d'assignació, cerca o addició. Per tant, estaríem parlant d'un total de $2n^2N$ productes en l'algoritme, el que implicaria una complexitat $O(n^2N)$, és a dir, quadràtica respecte al nombre d'estats i lineal respecte al nombre d'elements de la seqüència.

Determinació de μ

Les seccions anteriors han servit de base per finalment poder arribar en aquesta secció, que ens permetrà a partir d'unes dades d'entrenament (O fixat) trobar un model que expliqui el comportament d'aquesta. Formalment, volem trobar:

$$\arg\max_{\mu} P(O \mid \mu) \tag{4.35}$$

No existeix un mètode analític per determinar els paràmetres del model que maximitzin la versemblança $P(O \mid \mu)$, a causa de la complexitat associada al nombre exponencial de seqüències ocultes possibles. Aquesta complexitat es deriva del fet que $P(O \mid \mu)$ s'expressa com una suma sobre totes les seqüències possibles X, és a dir,

$$P(O \mid \mu) = \sum_{X} P(O, X \mid \mu),$$

on $P(O, X \mid \mu)$ inclou tant les probabilitats de transició com d'emissió. Com a resultat, no és possible obtenir una solució analítica directa amb tècniques clàssiques d'optimització.

No obstant, és possible trobar un màxim local de $P(O \mid \mu)$ utilitzant mètodes iteratius com l'algoritme Baum-Welch, el qual es basa en l'optimització iterativa com a un cas concret de l'algoritme EM (*Expectation Maximization.*). Aquesta aproximació resol el problema de manera iterativa mitjançant passos que alternen entre l'estimació dels paràmetres ocults i la maximització dels paràmetres del model.

Entrant en detall en l'algoritme, sigui $p_t(i, j)$ la probabilitat de transició entre l'estat i i l'estat j en un moment t determinat:

$$\begin{aligned} p_t(i,j) &= P(X_t = i, X_{t+1} = j \mid O, \mu) \\ &= \frac{P(X_t = i, X_{t+1} = j, O \mid \mu)}{P(O \mid \mu)} \\ &= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{n} \alpha_m(t)\beta_m(t)} \\ &= \frac{\alpha_i(t) a_{ij} b_{ijo_t} \beta_j(t+1)}{\sum_{m=1}^{n}\sum_{k=1}^{n} \alpha_m(t) a_{mk} b_{mko_t} \beta_k(t+1)} \end{aligned} \tag{4.36}$$

La primera igualtat és conseqüència de (4.31) , la tercera és a causa de la definició de $\beta_m(t)$ donada en l'algoritme 2 i la segona es segueix de (4.27) per al denominador i el numerador ve del fet que:

$$\begin{aligned} P(X_t = i, X_{t+1} = j, O \mid \mu) &= P(o_1, \ldots, o_{t-1}, X_t = i, X_{t+1} = j, o_t, o_{t+1}, \ldots, o_T \mid \mu) \\ &= P(o_1, \ldots, o_{t-1}, X_t = i \mid \mu) \cdot P(X_{t+1} = j, o_t, o_{t+1}, \ldots, o_T \mid o_1, \ldots, o_{t-1}, X_t = i, \mu) \\ &= \alpha_i(t) \cdot P(X_{t+1} = j, o_t, o_{t+1}, \ldots, o_T \mid X_t = i, \mu) \\ &= \alpha_i(t) \cdot P(o_{t+1}, \ldots, o_T \mid X_{t+1} = j, o_t, X_t = i, \mu) \cdot P(X_{t+1} = j, o_t \mid X_t = i, \mu) \\ &= \alpha_i(t) \cdot \beta_j(t+1) \cdot P(o_t \mid X_{t+1} = j, X_t = i, \mu) \cdot P(X_{t+1} = j \mid X_t = i, \mu) \\ &= \alpha_i(t) \cdot \beta_j(t+1) \cdot b_{ijo_t} \cdot a_{ij} \end{aligned}$$

On s'ha aplicat reiteradament (4.26) per anar desglossant les probabilitats i les definicions de cada terme. Cal fer especial incís en la cinquena igualtat, on s'ha usat que $P(o_{t+1}, \ldots, o_T \mid X_{t+1} = j, o_t, X_t = i, \mu) = P(o_{t+1}, \ldots, o_T \mid X_{t+1} = j, X_t = i, \mu)$ al ser la transició entre estats el que determina les futures observacions i no pas una observació anterior.

Seguint amb l'estimació del millor model, definim les variables $\gamma_i(t) = \sum_{j=1}^{n} p_t(i,j)$. Sumant respecte l'índex temporal t obtenim recomptes de valors que usarem com les nostres "esperances" o nombre de resultats esperats:

$$\sum_{t=1}^{N} p_t(i,j) \quad \text{(nombre esperat de transicions entre l'estat } i \text{ i l'estat } j\text{)} \tag{4.37}$$

$$\sum_{t=1}^{N} \gamma_i(t) \quad \text{(nombre esperat de transicions iniciades a l'estat } i\text{)} \tag{4.38}$$

Aquests seran els valors que usarem per a estimar el nostre nou model $\hat{\mu} = (\hat{A}, \hat{B}, \hat{\Pi})$ en cada iteració. No obstant, per començar, escollirem un model inicial $\mu_{=}(A, B, \Pi)$ fixat per tal de disposar d'un punt de partida. En cada iteració, els nostres paràmetres optimitzats seran:

$$\hat{\pi}_i = \gamma_i(1), \quad \text{freqüència esperada estat } i \text{ al moment } t = 1 \tag{4.39}$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{N} p_t(i,j)}{\sum_{t=1}^{N} \gamma_i(t)}, \quad \frac{\text{transicions esperades entre } i \text{ i } j}{\text{transicions esperades iniciades a } i} \tag{4.40}$$

$$\hat{b}_{ijk} = \frac{\sum_{\{t \mid o_t = k, 1 \leq t \leq N\}} p_t(i,j)}{\sum_{t=1}^{N} p_t(i,j)}, \quad \frac{\text{transicions esperades entre } i \text{ i } j \text{ on s'ha observat } k}{\text{transicions esperades entre } i \text{ i } j} \tag{4.41}$$

El model $\hat{\mu} = (\hat{A}, \hat{B}, \hat{\Pi})$ obtingut mitjançant aquest algoritme compleix que $P(O \mid \hat{\mu}) \geq P(O \mid \mu)$, segons va provar Baum [4].

En la pràctica, aquest procés no assegura trobar el millor model, sinó que sol acabar trobant un màxim local de la funció de versemblança $L(\mu \mid O) = P(O \mid \mu)$. Per aquest motiu, les condicions de parada de l'algoritme són establertes mitjançant un límit d'iteracions màximes i una tolerància ϵ que determina quan considerarem la diferència entre dos models "insignificant".

Existeixen diverses alternatives per a determinar la "distància" entre dos models generats $\mu^{(n)}$ i $\mu^{(n+1)}$ en iteracions de Baum-Welch. Generalment, els criteris de parada més usats són la diferència del logaritme de la funció de versemblança:

$$\mid \log P(O \mid \mu^{(n+1)}) - \log P(O \mid \mu^{(n)}) \mid < \epsilon \tag{4.42}$$

I la divergència KL:

$$D_{KL}(\mu^{(n)} \parallel \mu^{(n+1)}) < \epsilon = \sum_{i,j} a_{ij}^{(n)} \log \frac{a_{ij}^{(n)}}{a_{ij}^{(n+1)}} + \sum_{i,j,k} b_{ijk}^{(n)} \log \frac{b_{ijk}^{(n)}}{b_{ijk}^{(n+1)}} + \sum_{i} \pi_{i}^{(n)} \log \frac{\pi_{i}^{(n)}}{\pi_{i}^{(n+1)}} < \epsilon \tag{4.43}$$

El primer sol ser el més usat per la seva senzillesa, ja que com s'ha vist anteriorment, existeixen algoritmes específics per a poder calcular eficientment la probabilitat d'una observació donada un model. El logaritme de la funció de versemblança s'utilitza per evitar els problemes d'*underflow* (valors per sota de la precisió de la màquina) i permet capturar de manera més eficient la fluctuació entre probabilitats segons la seva magnitud (per exemple, $\mid log(0.02) - log(0.01) \mid$ captura millor l'impacte del canvi que $\mid 0.02 - 0.01 \mid$).

La divergència KL és útil per la seva naturalesa que ofereix una comparació directa entre les probabilitats de cada model, però per a matrius amb dimensions elevades pot resultar computacionalment ineficient.

4.4 Aplicació dels models a la traducció automàtica

Un cop tenim els models per a la llengua d'origen i destí, la traducció automàtica es basa en un mètode per determinar quina és la millor traducció possible i com de precisa és. Un dels enfocaments més comuns és el model del canal sorollós, que busca maximitzar $P(e \mid f)$, és a dir, la probabilitat que una seqüència en la llengua destí (e) sigui la traducció d'una seqüència en la llengua d'origen (f). Amb el teorema de Bayes, aquest problema es pot reformular com $P(e \mid f) \propto P(f \mid e)P(e)$, on $P(f \mid e)$ és el model de traducció que ens diu quina probabilitat té una seqüència de la llengua d'origen de generar una seqüència en la llengua destí, i $P(e)$ és el model de llengua, que mesura la fluïdesa de la seqüència en la llengua destí.

Els models ocults de Màrkov i els n-grames són claus en aquest enfocament. Els n-grames s'utilitzen per modelar $P(e)$, basant-se en la probabilitat de seqüències de paraules segons la seva freqüència en grans corpus lingüístics. D'altra banda, els models ocults de Màrkov s'apliquen a $P(f \mid e)$, on ens ajuden a establir les alineacions entre les paraules de l'origen i la destí a través d'estats ocults, juntament amb les probabilitats d'emissió i de transició.

L'alineació de paraules és un element fonamental en aquest procés, ja que ens permet saber com cada paraula de la llengua d'origen es correspon amb les paraules en la llengua destí. En el context del model del canal sorollós, l'alineació és clau per calcular $P(f \mid e)$, ja que aquesta probabilitat depèn directament de la correspondència entre les paraules en els dos idiomes. Els models ocults de Màrkov són molt útils per fer aquesta alineació, ja que permeten modelar la seqüència de transicions entre estats ocults (les paraules en la llengua destí) i les paraules observades en l'idioma d'origen. Amb les probabilitats de transició, es poden capturar patrons lingüístics, com l'ordre sintàctic, mentre que les probabilitats d'emissió ajuden a determinar quin terme és més probable per a cada paraula d'origen.

Finalment, l'entropia creuada s'utilitza sovint per mesurar quan bé s'adapta un model de traducció a les dades. Aquesta mesura compara les probabilitats predites pel model amb les observades en un corpus de validació. L'entropia creuada calcula el cost mitjà d'utilitzar el model μ per predir les observacions O, i es defineix com:

$$H(P, Q) = -\sum_{x} P(x) \log Q(x),$$

on $P(x)$ és la distribució real i $Q(x)$ la distribució predita pel model. Alternativament, donat que la distribució real no és possible determinar-la, fem servir l'aproximació (3.21) Un valor baix d'entropia creuada indica que el model està fent bones prediccions, mentre que un valor alt indica que el model no s'adapta bé. Aquesta mesura és fonamental a l'hora de validar i entrenar el model, ja que permet ajustar els paràmetres per minimitzar la diferència entre les prediccions i les dades reals, millorant així la precisió de la traducció. En traducció automàtica , l'entropia creuada és útil per comparar diferents models i decidir quin s'ajusta millor a les dades.

5 Models Neuronals i Traducció Automàtica

Els models de traducció automàtica han experimentat una evolució fascinant al llarg de les darreres dècades. Inicialment, la traducció es basava en models estadístics que utilitzaven grans bases de dades de textos paral·lels per calcular les probabilitats de correspondència entre paraules i expressions en diferents idiomes. Aquests models estadístics, coneguts com a SMT (Statistical Machine Translation), intentaven predir la millor traducció fent ús de tècniques estadístiques i modelacions del llenguatge per tal d'obtenir els seus resultats.

L'arribada de les xarxes neuronals i el *Deep Learning* ha comportat canvis transcendentals en el camp de la traducció automàtica, canviant per complet la mentalitat i metodologia utilitzades fins el moment. Els nous models de traducció neuronal (NMT, Neural Machine Translation) aprofiten l'arquitectura de xarxes neuronals profundes per aprendre representacions més complexes i contextuals dels idiomes. Aquests models són capaços d'entendre millor les relacions entre paraules i capturar el significat de frases completes, aconseguint traduccions més naturals i precises. Aquest avanç ha situat la traducció automàtica en un nivell de qualitat superior, desbancant en molts aspectes als models estadístics anteriors.

5.1 Sequence to Sequence Model

El primer model que podem considerar cabdal per al desenvolupament del NMT és el model *Sequence to Sequence model with attention*[21] (Seq2Seq model) introduit l'any 2014, que feia ús de l'arquitectura *Encoder-Decoder* per tal de codificar els inputs a vectors de llargada fixa token per token; a la vegada que actualitza els valors d'un vector de çontext"per tal de intentar identificar el "sentit"de l'input. Aleshores el decoder produeix seqüencialment per tokens la seqüència en l'idioma objectiu originada per l'input, tenint en compte el vector de context de l'encoder i els token que hagi generat prèviament. En aquest model en concret, tant decoder com encoder eren xarxes neuronals recurrents (RNN) que permetien una millor traducció degut a la capacitat de tenir en compte els resultats de passos previs.

El problema principal del model Seq2Seq era el fet que utilitzava una codificació de llargada fixa per a les seqüències d'entrada, donant lloc així a problemes i traduccions poc acurades quan s'introduíen frases de gran llargada.

5.2 Neural Machine Translation by Jointly Learning to Align and Translate

L'any 2015 Bahdanau et.al [3] introduiren l'anomenat mecanisme d'atenció per tal de fer front a les principals limitacions del model Seq2Seq.

Tot i que el nou model introduït seguia estant basat en l'arquitectura encoder-decoder, presentava importants diferències, que permetien tenir en compte tot el context de la frase.

Per començar, l'encoder feia servir una Bidirectional Recurrent Neural Network (BRNN). La diferència principal entre aquesta i una senzilla, es que el context de cada paraula és processat dos cops, en una tenint en compte la informació del token d'entrada previ ("forward pass") i l'altra centrant-se en el token d'entrada posterior (backward pass), combinant ambdòs (mitjançant alguna operació com suma, concatenació) en un únic resultat que determinava el *hidden state* de l'encoder en cada pas . Tot i que aquesta operació augmenta el cost computacional de la xarxa, resulta extremadament útil en tasques relacionades amb el llenguatge, ja que permet tenir una visió més general de l'input.

És la introducció d'aquest doble processament el que permet introduir l'anomenat "mecanisme d'atenció"que determina en quina part de l'input s'ha de "fixar"(en aquest cas, entés com donar més relevància) el decoder per determinar el resultat del pas corresponent.

Aquest mecanisme d'atenció consistia en associar una puntuació o *alignment score* entre cada element i de l'input i cada j element de l'output a ser generat. Aquesta era calculada a partir del hidden state de l'encoder en el pas actual i del hidden state de l'encoder en el pas anterior. Si considerem t_{ij} la alignment score entre l'element i d'entrada i el j de sortida:

$$t_{ij} = alignment\, score(s_j - 1, h_i) \tag{5.1}$$

Un cop obtenides totes aquestes puntuacions, es transformaven amb una funció softmax per obtenir les *attention weights* o "pesos d'atenció" a_{ij}:

$$\alpha_{ij} = \frac{e^{t_{ij}}}{\sum_k e^{t_{kj}}} \tag{5.2}$$

A partir dels pesos d'atenció, es generava una suma ponderada amb els hidden states h_i de cada element d'entrada de l'encoder per obtenir un vector de context per a cada j paraula del target a ser generada:

$$c_j = \sum_i \alpha_{ij} h_i \tag{5.3}$$

Finalment, a partir del vector de context i l'actual hidden state del decoder, es generava la paraula target j buscada.

D'aquesta manera s'eliminava la necessitat de codificar tot l'input amb un vector de llargada fixa, sinó que s'introduia un hidden state particular per a cada element i d'aquest. Així, el model presentava una millora notable respecte al Seq2Seq, al permetre treballar més extensament amb inputs de major complexitat gramatical i llargada.

Aquest mecanisme permetia a la xarxa neuronal determinar dinàmicament quina de les parts de la frase d'entrada havia de tenir mes rellevància a l'hora de calcular el nou output, el que permet donar resultats més precisos i naturals.

Malgrat la millora dels resultats obtinguts, aquest mecanisme d'atenció importava un augment notable del cost computacional del model, donada la introducció del backward pass i del vector de context.

5.3 Transformer Model

La introducció del mecanisme d'atenció en el model descrit en la secció anterior va donar lloc no només a una millora significativa en els models NMT, sinó també a l'inici d'un canvi de paradigma que consolidaria definitivament l'hegemonia dels models neuronals de traducció sobre els estadístics.

Aquest fet es consolidaria amb la introducció del model *Transformer* o Transformador per Vaswani et.al [22] l'any 2017, que proposaria un model de traducció basat çompletament en l'atenció".

Fins la introducció d'aquest, els models basats en xarxes neuronals recurrents amb mecanismes d'atenció eren la base de les tasques de modelatge seqüencial del llenguatge i traducció [5], i part de la recerca estava enfocada en sobrepassar els límits dels models del moment per tal de millorar la seva eficàcia mitjançant l'exploració de tècniques de factorització [17] o mitjançant *conditional computation* [20], procediment que consisteix en activar només certes parts del model segons l'input introduït.

El canvi radical proposat en el transformador consistia en eliminar completament l'ús de xarxes neuronals recurrents, per centrar-se en usar únicament mecanismes d'atenció, fet que permetia la paral·lelització del model i la capacitat de capturar la relació entre elements de l'input independentment de la seva posició. Per tal d'aconseguir aquestes característiques, es van introduir dos nous mecanismes d'atenció, el *self-attention mechanism* i el *multi-head attention*.

Donada la limitació d'espai, l'Àpendix 1 ofereix una anàlisi teòrica detallada de l'estructura del model transformador i del funcionament dels mecanismes d'atenció que introdueix. També inclou detalls sobre la implementació d'una variant d'aquest, *MarianMT*, seguint pràctiques estàndards en la industria i el món acadèmic d'implementació d'aquests models avui en dia.

L'apendix és per tant essencial per a obtenir una comprensió més profunda dels models neuronals i es recomana vehementment la seva consulta.

5.4 Importància del Transformer

La introducció del model Transformer va suposar un canvi de paradigma en el referent a la traducció automàtica, aconseguint els millors resultats fins el moment en el benchmark WMT 2014 (Workshop on Machine Translation), una sèrie de proves i datasets ampliament usats en el món de la traducció automàtica per evaluar l'eficiència dels models. Aquesta millora en els resultats també va venir acompanyada també d'una millora en l'eficiència a l'hora de l'entrenament del model, ja que requeria menys recursos computacionals per aconseguir-ho, degut a la capacitat de paral·lelització dels càlculs al eliminar la seqüencialitat dels models basats en RNNs. Per altra banda, també introduïa la capacitat d'obtenir encara millors resultats mitjançant l'escalabilitat del model original.

De fet, la seva aparició va ser crucial per al desenvolupament no només de la traducció automàtica, sinó també del NLP en general amb l'aparició dels models multimodals pre-

entrenats, com BERT (Bidirectional Encoder Representations from Transformers) [11] i GPT (Generative Pre-trained Transformer) [1], sent aquest últim la base d'alguns chatbots famosos a nivell mundial, com ChatGPT.

6 Conclusions

Aquesta memòria ofereix una introducció a diversos models estadístics i neuronals aplicats a la traducció automàtica. A més d'explicar-ne els fonaments teòrics, també inclou la implementació pràctica d'un d'aquests models, mostrant com, amb el temps, no només s'han desenvolupat models més complexos i precisos, sinó que també s'ha facilitat l'accés a aquestes tecnologies per a tots els usuaris.

El treball destaca el caràcter interdisciplinari de la traducció automàtica, mostrant com les matemàtiques es poden aplicar a àmbits tan diversos com la lingüística. Aquesta connexió entre tècniques numèriques i llenguatges aparentment desconnectats (el tècnic i el natural) evidencia el paper fonamental de les matemàtiques en la descripció i predicció de tasques relacionades amb les humanitats.

A més, es presenta la informàtica com un pont essencial entre la recerca teòrica i les aplicacions pràctiques, reforçant la seva importància com a eina clau en la resolució de problemes.

El camp de la traducció automàtica és immens, amb una gran diversitat de tècniques i teories que no es poden abordar en la seva totalitat en una única memòria. Tot i així, aquest treball posa en relleu com aquesta disciplina continua avançant i trencant barreres, facilitant la comunicació entre persones, una de les funcions fonamentals per a la interacció social.

Durant la realització d'aquest treball, s'ha aprofundit en àrees de les matemàtiques que fins ara no s'havien explorat (com la teoria de la informació) i s'han investigat els fonaments darrere de les solucions informàtiques pràctiques. Aquesta experiència ha estat enriquidora, ja que m'ha permès comprendre tant la complexitat com la importància de l'aplicació dels coneixements i bases de raonament adquirits durant el grau.

Algunes possibles extensions del treball serien : implementació des de 0 del model Transformador, profundització en les tècniques de traducció automàtica (per exemple, l'alineament de textos), introduir una nova secció amb tècniques per modelar gramàtiques i sintaxis de manera explícita i implementació d'un model ocult de Màrkov entrenat mitjançant l'algoritme de Baum a partir d'un corpus escollit.

Apèndix A

Detalls teòrics del model Transformer

Com s'ha explicat en la secció corresponent, el model Transformador va resulta revolucionari per eliminar completament la necessitat de tenir processaments seqüencials i passar a tractar els problemes de traducció com un problema complet d'atenció (d'aquí el títol de l'article *Attention is all you need*) mitjançant la introducció de dos mecanismes : *self-attention mechanism* i el *multi-head attention*.

En aquesta secció s'ofereixen els detalls exactes d'amdues tècnics i de com s'ensamblen a l'arquitectura bàsica *Encoder-Decoder* del model Transformador.

Estructura general

L'estructura general del *Transformer* segueix el framework bàsic Encoder-Decoder com la resta de models presentats anteriorment.

La idea principal d'aquesta arquitectura és entrenar l'encoder per tal que aquest obtingui representacions codificades de l'input que després són decodificades pel decoder per obtenir una traducció de la frase d'entrada.

Cada un dels components està organitzat en N capes (original i habitualment $N = 6$) on cada capa consta de 2 subcapes principals, cada una seguida per una capa d'addició residual i normalització:

- Subcapa *Multi-head Self Attention*
- Subcapa *Feed Forward Neural Network* (FFN)

Abans d'entrar en detall en el funcionament de cada component i les seves subcapes cal fer un petit incís sobre com representem els inputs rebuts pel model.

Representació d'inputs

Els inputs són dividits en tokens o *símbols* (generalment paraules en problemes de traducció automàtica) que després són representats per vectors de llargada fixa anomenats *embeddings*, de manera similar al model Seq2Seq i allunyant-se així de les tècniques usades en el moment.

Així doncs, cada paraula del nostre vocabulari (o conjunt de tokens) la representem amb un vector de llargada o dimensionalitat fixa d. Per tant, si V és el nombre d'elements del vocabulari, podem definir la nostra aplicació d'embedding E com:

$$E : \mathbb{R}^V \to \mathbb{R}^d \tag{A.1}$$

Aquesta aplicació determina una matriu de mida $V \times d$, que representarà els paràmetres a optimitzar del nostre encoder.

L'encoder és entrenat justament per intentar produir els embeddings més acurats possibles que aconsegueixin capturar el significat de les paraules, per a que d'aquesta manera

els sinònims acabin obtenint valors molt similars entre si i els antònims siguin radicalment diferents.

Generalment aquestes representacions és realitzen mitjançant vectors densos d'alta dimensionalitat, és a dir, vectors amb valors diferents de 0 i un nombre de components elevats. Per exemple, en el Transformer original proposat per Vashwani et.al la dimensionalitat dels embeddings era de 512 components.

Un dels avantatges d'aquest mètode de codificació és que s'introdueix la possibilitat de fer els càlculs de cada embedding de manera simultània, introduint així la capacitat de fer ús de computació paral·lela per tal d'obtenir una major eficiència en comparació amb els models basats en BRNN.

A priori, aquesta tècnica de representació no inclou informació sobre la posició de la paraula dins de la frase, motiu pel qual el transformer feia una modficació de l'embedding obtingut per donar implícitament en aquest una idea no només de la semàntica sinó també de la posició, permetent així treballar amb significat i context alhora. Aquesta modificació rep el nom de *positional encoding*.

Positional encoding

El procés de positional encoding es basa en afegir a l'embedding del token obtingut una funció determinada per la posició del token dintre de l'input, la dimensió dels embeddings i de les funcions trigonomètriques sinus i cosinus.

Aquesta funció no introdueix cap paràmetre nou, de manera que resulta ideal per reduir la complexitat del model i facilitar així el seu entrenament; no obstant, segueix permetent incloure informació del lloc que ocupa el token dintre l'output.

Per a cada posició *pos* del token en l'encoding, d la dimensió de l'embedding, i l'índex de posició i dintre del rang $[0, (d/2) - 1]$, el vector de positional embedding del token a la posició *pos* (PE_{pos}) es calcula component a component de la següent manera:

$$\begin{aligned} PE_{(pos,2i)} &= sin(\frac{pos}{10000^{\frac{2i}{d}}}) \\ PE_{(pos,2i+1)} &= cos(\frac{pos}{10000^{\frac{2i}{d}}}) \end{aligned} \tag{A.2}$$

Hi ha diversos motius pels quals s'escullen aquestes funcions en específic:

- Les components de baix índex varien lleugerament entre posicions, mentre que les d'alt índex difereixen en gran mesura. D'aquesta manera, part del vector de posició captura els detalls entre els diversos tokens, mentre que l'altra es dedica a informar sobre els trets distintius a gran escala entre els elements de l'input.
- La continuïtat de les funcions sinus i cosinus permeten no restringir-se en una longitud fixada, així el model manté una eficàcia extensible a inputs de diferent mida amb els que va ser entrenat, proveint així un avantatge sobre els models on els positional encodings són parametritzats i optimitzats durant l'entrenament, donant així lloc a *overfitting* en frases de llargada igual o similar a les usades en l'aprenentatge.
- Donat que la diferència per a cada component de un vector de posició només dependrà de la diferència entre les posicions dels tokens, es permet capturar correctament la distància relativa entre els elements de l'input. No obstant cada índex del

vector PE serà únic de manera que també s'inclou la informació sobre la posició absoluta de cada element en l'entrada. Així doncs aquestes funcions permeten inferir de manera completa la jerarquia de posicions tan absoluta com relativa de les dades d'entrada.

Una vegada entesa la representació dels inputs entrats, podem passar a analitzar profundament les dos parts principals del transformador: l'encoder i el decoder.

Encoder

Com s'ha explicat anteriorment, l'encoder és la part que es dedica en representar els inputs de manera que segueixin mantenint la informació semàntica i contextual que tenien en la seva forma original. Per tal de fer-ho, es construeix un bloc de 6 capes, on cada capa està alhora conformada per 3 subcapes.

La primera subcapa és l'anomenada *Multihead Self-Attention layer* que permet al model extreure la informació codificada sobre les posicions relatives i absolutes entre els tokens, independentment de la distancia entre elles. Les tasques portades a terme en aquesta capa per tal d'interpretar les relacions entre els elements de l'entrada rebuda són les següents.

Per tal de fer-ho el model primerament calcula una attention score per a cada element d'entrada, però d'una manera molt diferent a la presentada per Bahdanau et.al.

Primerament, tot embedding rebut pel mecanisme d'entrada es separat en 3 vectors diferents, anomenats Q (Query), K (Key) i V (Value). Cada un d'aquests captura un aspecte diferent del que resulta ïmportant"del token rebut:

- Query : Determina en quines parts de l'input s'ha de centrar el token rebut per tal de complementar la informació que ja aporta aquest. Es pot entendre com una manera d'assignar un pes o valor a cada token diferent del processat.
- Key : Representa les característiques clau del token, és a dir, la informació que pot resultar útil per als altres elements de la seqüència d'entrada. Actua com un recíproc de Query, ja que aquest indica la importància de la resta de tokens per al processat, mentre que Key representa quan útil pot ser el token processat per a la resta.
- Value : Representa la informació que aportarà el token en cas de que aquest hagi rebut "atenció". En contraposició a Key i Query que s'usen per determinar els pesos d'atenció, Value és el vector que conté el significat i context que es vol interpretar. Són les dades rellevants per a l'output.

Cada una d'aquestes components es calculada a partir de l'embedding X mitjançant aplicacions lineales representades per matrius de pesos. Així doncs:

$$Q = XW^Q$$

$$K = XW^K$$

$$V = XW^V$$

On W representa la matriu de pesos de cada component de l'atenció segons el seu índex superior. Els valors d'aquestes matrius són determinades durant l'entrenament del model, i tenen una dimensionalitat de (TODO:Afegir aqui un estudi del a dimensionalitat).

Un cop determinades les tres components, es procedeix a calcular l'atenció donada a cada token mitjançant els productes escalars dels vectors Query i Key. Així doncs, per a cada element i de l'input i cada token diferent j, es calcula la puntuació o score entre ambdós de la següent manera:

$$score_{i,j} =< Q_i, K_j > \tag{A.3}$$

On $< \cdot, \cdot >$ representa el producte escalar entre dos vectors.

És fàcil entendre que aquest producte escalar resulta una manera de determinar la magnitud de la compatibilitat o relació entre dos vectors donats, al representar Q_i i K_j dades reciproques que es complementen entre si, ja que Q_i indica quina es la importància de la resta dels elements de l'input per al token en la posició i i K_j representa la utilitat de l'element j per a la resta dels tokens d'entrada.

Un cop calculada l'score aquesta és escalada amb la dimensionalitat dels vectors Query i Key per evitar l'aparició de valors desmesurats que puguin esbiaixar els càlculs d'atenció.

Aleshores, si d_K és la dimensionalitat del vector Key (que coincideix amb la del vector Query), escalem l'score de la següent forma:

$$scaled\, score_{i,j} = \frac{score_{i,j}}{\sqrt{d_K}} = \frac{< Q_i, K_j >}{\sqrt{d_K}} \tag{A.4}$$

Finalment, apliquem una funció softmax per transformar els valors obtinguts en una distribució de probabilitat, que ens determinin els pesos d'atenció entre l'element i del token i cada un de la resta dels tokens j de la seqüència d'entrada:

$$attention\, weight_{i,j} = \frac{e^{scaled\, score_{i,j}}}{\sum_{k=1}^{d_K} e^{scaled\, score_{i,k}}} \tag{A.5}$$

Finalment, per calcular l'atenció es realitza la suma ponderada per l'atenció dels vectors Value:

$$Attention(Q, K, V) = \sum_{j=1}^{n} attention\, weight_{i,j} \cdot V_j \tag{A.6}$$

D'aquesta manera, l'output del mecanisme d'atenció per a cada token serà una mitja ponderada del valor (o ïnformació real"que aporta cada token al d'entrada.

Aquest mecanisme d'atenció presenta dos avantatges principals:

Primerament, la separació en tres vectors de la rellevància que ha de tenir un element de la seqüència permet afinar molt més la importància de cada token depenent segons la seva posició i la informació semàntica i contextual que aporta.

D'aquesta manera, es mira per separat que és el que realment aporta l'element d'entrada al significat (Value), quan fortament està relacionat amb cada un dels altres elements (Query) i què és el que pot aportar a la resta d'elements (Key). Aquesta separació entre contingut i relació aporta al model la capacitat de ser capaç de captar de manera més

acurada l'entrada a nivell general, donant lloc a millors resultats.

Per altra banda, donats que el càlcul de l'atenció per a cada vector no depèn seqüencialment de la resta d'elements, es poden aplicar tècniques de computació paral·lela per millorar l'eficiència del model. Bàsicament, com que el càlcul de Q,K i V de cada vector només necessita aquest, podem realitzar els càlculs simultàniament d'aquests (i per tant, de l'atenció també) simultàniament.

L'estructura descrita anteriorment representaria una *head* del mecanisme d'atenció. Les subcapes de *multihead attention* presenten diverses heads (8 en el model original) treballant alhora per tal de capturar diversos aspectes de les relacions entre els tokens.

Cada una d'aquestes genera el seu propi output que més tard es concatenat per a donar el resultat que serà enviat a la següent capa. Per tal de combinar-les, s'aplica la concatenació dels resultats de les diverses heads, i a continuació són transformades per una aplicació lineal determinada per una matriu W_0, els coeficients de la qual són apresos durant l'entrenament del model.

Així doncs, si n és el nombre de *heads* i H_i representa la i-èssima *head*; l'output d'una subcapa de Multihead Self-Attention seria:

$$Output\,(Q, K, V) = Concat(H_1, H_2, .., H_n)W_0 \tag{A.7}$$

Aquest procés s'ha descrit a nivell vectorial per a facilitar la seva comprensió, no obstant, en la pràctica, es computen matrius senceres per tal d'assegurar la màxima eficiència computacional possible.

Aquest resultat és aleshores introduït en una FFN (*Feed Forward Neural Network)* consistent en dos capes completament conectades entre si amb una capa ReLU entre elles, (*Rectified Linear Unit*) que aplica la següent transformació a cada token rebut de la subcapa d'atenció :

$$FFN(x) = max(0, x_1W_1 + b_1)W_2 + b_2 \tag{A.8}$$

On W_i indica una matriu de pesos i b_i un vector de *bias* de la FFN. Cal destacar que l'aplicació lineal definida per W_1 genera un vector de major dimensionalitat que la dels embeddings donats fins el moment, mentre que W_2 redueix aquesta dimensionalitat a l'original.

Aquesta capa és introduida pels següents motius:

1. Introduir comportament no lineal per tal de facilitar l'aprenentatge de patrons complexos mitjançant la capa ReLU

2. Extreure més característiques de les representacions resultants del mecanisme d'atenció per tal de captar més matissos entre les relacions i significats dels tokens. Això s'aconsegueix gràcies a la major dimensionalitat de W_1. Per exemple, en el transformador original W_1 era una matriu de mida 512×2048, generant així un vector de dimensió 1×2048

3. Realitzar una extracció de les característiques més importants determinades a l'augmentar la dimensionalitat reduint el vector a la dimensió orignal dels embeddings. Això s'aconsegueix reduint la mida de W_2, com es pot veure en el model original, on era una matriu 2048×512.

Finalment, cal afegir que per a cada subcapa en l'encoder s'afegeix una altra subcapa d'addició residual (l'input de cada capa és sumat a l'output d'aquesta)[15] i normalització [2] per tal de facilitar l'entrenament del model.

Decoder

Abans d'entrar en detall en l'estructura del decoder, és necessari recordar que per tal d'entrenar el Transformer, es va proposar que el decoder no només rebés les dades de l'encoder com a input, sinó que durant aquest període rebés la frase de target traslladada una posició a la dreta. Per altra banda, en la fase de predicció, aquest rep els tokens que ha generat prèviament. Aquestes dades són combinades amb les de l'encoder més tard en el procés.

Anàlogament a l'encoder, el decoder està composat per N capes (6 en el Transformer original) on cada capa està composada per 3 subcapes.

La primera subcapa és una *Masked Multi-Head Self-Attention layer*, és a dir, una capa molt similar a la de l'encoder per tal d'atendre al seu input (d'aqui que sigui *self-attention*). Aquesta capa està emmascarada per tal que no tingui en compte l'informació generada en tokens futurs.

Notis que en la secció anterior hem explicat el procés de càlcul de l'atenció vectorialment, però s'ha fet l'incís que en la pràctica es realitza mitjançant matrius. En aquesta secció, s'usaran els càlculs matricials per representar millor com funcionaria realment.

Similarment a l'encoder, l'input són els embeddings dels tokens corresponents, i de cada un d'ells se'n calculen les tres matrius Query,Key i Value, calculades per les matrius de pesos W^Q, W^K, W^V respectivament. Si X és la matriu que on cada i-èssima és l'embedding de l'i-èssim token de la seqüència d'entrada tenim :

$$Q = XW^Q$$

$$V = XW^V$$

$$K = XW^K$$

A continuació, es calcula la matriu de puntuacions d'alineament escalades A, on cada element A_{ij} representa l'alignment score entre els elements i,j de l'input. Si Q és la matriu de queries de tots els elements, K la matriu de keys de cada element, i d_K la dimensió de cada vector Key (que coincideix amb la de cada vector Query), tenim:

$$A = \frac{QK^T}{\sqrt{d_K}} \tag{A.9}$$

A diferència de l'encoder, el decoder només té en compte els elements generats fins al moment, es per això que en aquest pas s'aplica l'emmascarament. Per tal de fer-ho, es genera la matriu de puntuacions emmascarades M per a cada i-èssim element a generar de l'output, definida de la següent manera:

$$M_{ij} = \begin{cases} A_{ij} \text{ si } i < j \\ -\infty \text{ si } i \geq j \end{cases} \tag{A.10}$$

A continuació s'aplica la funció softmax en cada fila d'aquesta matriu d'emmascarament, per generar la matriu de pesos d'atenció, que anomenarem T; i generem la matriu d'atenció, on cada fila i indicara el vector d'atenció (l'output resultant) de cada i-èssim element de l'input:

$$Output = TV \tag{A.11}$$

On recordem que V és la matriu on cada i-fila és el vector de valor de l'i element de la seqüència d'entrada.

De nou, cada una d'aquestes estructures representaria una *head* del mecanisme d'atenció. En el Transformer original, el decoder constava de 8 heads, on cada una d'elles "parava atenció"en una característica diferent del que seria l'output.

D'aquesta manera s'aconseguia representar millor les diverses relacions que poden existir entre els tokens d'entrada. Per exemple, una de les matrius d'atenció generades pot estar centrada en les relacions sintàctiques (o com les normes gramaticals de la llengua imposen que els elements s'agrupin entre si) i una altra en les semàntiques (quin significat real s'aporta a l'oració).

Aquest exemple no implica que necessàriament sigui així, sinó que pretén donar a entendre el significat de "parar atenció"en el Transformer com una manera de determinar el valor dels aspectes que conformen una construcció tan complexa com és una llengua.

La següent subcapa que constitueix el decoder és la *Multi-head Cross-Attention*. Es tracta d'una capa molt similar a l'anterior, amb la diferència que en aquest cas l'input que rep és l'output de l'encoder. D'aquesta manera es consegueix combinar l'informació obtinguda fins el moment pel decoder i l'obtinguda per l'encoder. Així doncs, es torna a calcular l'atenció de manera similar, però en aquest cas, Queries, Keys i Values són determinats de la següent forma:

$$Q = X_{decoder}W^Q$$

$$K = X_{encoder}W^K$$

$$V = X_{encoder}W^V$$

On les W indiquen les matrius de pesos determinades durant l'entrenament i el subfix de la X indica si l'output prové de la capa anterior del decoder o és el l'output final de l'encoder.

I calculem l'atenció creuada (o combinada) com :

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_K}})V \tag{A.12}$$

A continuació s'usen les mateixes subcapes que en l'encoder, amb una FFN que rep directament l'atenció calculada, l'objectiu de la qual es també augmentar i comprimir la dimensionalitat i afegir no linearitat.

Finalment, al resultat de la subcapa anterior s'hi afegeix la connexió residual de l'input i es normalitza el resultat obtingut.

A continuació, es presenta un gràfic descrivint l'arquitectura del Transformer original de manera visual:

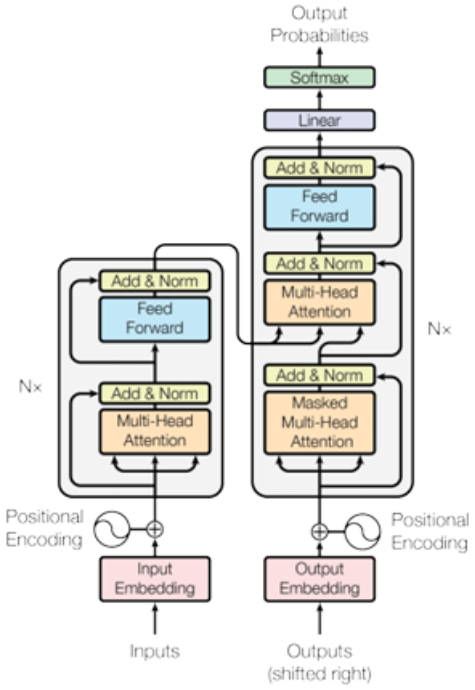

Figura 5: Arquitectura del Transformer. Extret de [22]

Detalls de la implementació

Com a part pràctica i per a il·lustrar un *pipeline* d'entrenament d'un model basat en el transformador s'ha decidit entrenar el model preimplementat *MarianMT* amb el dataset *Europarl* per a la tasca de traducció de l'anglès a l'espanyol.

La implementació s'ha dut a terme en un notebook de python i un entorn d'execució de Google Colab, específicament amb el $T4 - GPU$, donat a limitacions en la disposició de hardware a nivell personal. Diverses llibreries son necessàries, però poden ser instal·lades executant la primera cel·la de codi del notebook. Destacar especialment les llibreries de Hugging Face [13], que han proveït el model *MariaMT* preimplementat, l'API d'entrenament, el tokenitzador de texts i el dataset Europarl.

Els motius pels quals s'ha decidit fer ús de les llibreries de Hugging Face per a les tasques fonamentals de construcció i entrenament del model es discuteixen en el mateix notebook, però es poden resumir en: disponibilitat, eficiència i facilitat d'ús.

En el notebook disponible al codi font s'inclou una explicació detallada pas a pas de les diverses etapes d'entrenament del model: preprocessament de dades, implementació del model, entrenament i avaluació.

A continuació s'ofereix un breu resum del procediment seguit:

1. Preprocessament de les dades: Càrrega de l'Europarl, tokenització de les dades obtingudes, separació en *splits* d'entrenament i validació.
2. Implementació : Càrrega del model preentrenat *MarianMT*, configuració dels hiperparàmetres, ús de l'optimitzador ADAM i la entropia creuada com a funció de pèrdua.
3. Entrenament: Processament de les dades del dataset i actualització dels paràmetres del model.
4. Avaluació : Càlcul de la BLEU score per determinar l'efectivitat del model. Aquesta mètrica d'avaluació ha estat escollida al ser usada de manera recurrent en la indústria (de fet, va ser usada per determinar l'eficiència del Transformer original introduït el 2017).

Per a replicació dels resultats i un anàlisi més detallat sobre el procediment seguit es disposa del notebook.

L'objectiu d'aquesta implementació era oferir una visió pràctica dels conceptes teòrics explicat seguint procediments habituals d'entrenament de models per a traducció automàtica, en cap moment es pretenia fer un estudi sobre optimització d'hiperparàmetres o una implementació personalitzada del transformador.

Possibles extensions d'aquesta implementació serien:

- Comparació del rendiment entre diversos models. Per exemple, fer ús del model BERT per observar les diferències de rendiment entre un model general i un específic per a traducció.
- Experimentació amb hiperparàmetres :Analitzar com varien els resultats en funció de factors com el nombre d'epochs, el learning rate i el batch size.. Intentar trobar empíricament una combinació que resulti relativament òptima.

- Ús d'altres datasets: Seria interessant comprovar quin rendiment obté *MarianMT* quan es entrenat amb corpus de major dimensió com per exemple el corpus OPUS, que és una recopil·lació de datasets usats en processament del llenguatge natural.

Bibliografia

[1] Christoph Alt, Marc Hübner i Leonhard Hennig. *Fine-tuning Pre-Trained Transformer Language Models to Distantly Supervised Relation Extraction*. 2019. arXiv: 1906.08646 [cs.CL]. URL: https://arxiv.org/abs/1906.08646.

[2] Jimmy Lei Ba, Jamie Ryan Kiros i Geoffrey E. Hinton. *Layer Normalization*. 2016. arXiv: 1607.06450 [stat.ML]. URL: https://arxiv.org/abs/1607.06450.

[3] Dzmitry Bahdanau, Kyunghyun Cho i Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL]. URL: https://arxiv.org/abs/1409.0473.

[4] Leonard E. Baum et al. "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains". A: *The Annals of Mathematical Statistics* 41.1 (1970), pàg. 164 - 171. ISSN: 00034851, 21688990. URL: http://www.jstor.org/stable/2239727 (cons. 09-01-2025).

[5] Denny Britz et al. "Massive Exploration of Neural Machine Translation Architectures". A: *CoRR* abs/1703.03906 (2017). arXiv: 1703.03906. URL: http://arxiv.org/abs/1703.03906.

[6] Zdzisław Brzeźniak i Tomasz Zastawniak. *Basic Stochastic Processes: A Course Through Exercises*. Springer Undergraduate Mathematics Series. London: Springer, 1999. ISBN: 978-1-4471-0533-6. DOI: 10.1007/978-1-4471-0533-6. URL: https://link.springer.com/book/10.1007/978-1-4471-0533-6.

[7] George Casella i Roger L. Berger. *Statistical Inference*. 2nd. Pacific Grove, CA: Duxbury, 2002. ISBN: 978-0534243128.

[8] Institut d'Estudis Catalans. *Corpus Textual Informatitzat de la Llengua Catalana*. Recuperat: Novembre 13, 2024. n.d. URL: https://ctilc.iec.cat/scripts/IniPresentacio.asp.

[9] Thomas M. Cover i Joy A. Thomas. *Elements of Information Theory*. 2nd. Hoboken, NJ, USA: Wiley-Interscience, 2006. ISBN: 978-0-471-24195-9.

[10] Kenneth Ward Church i William A. Gale. "A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams". A: *Computer Speech & Language* 5 (1991), pàg. 19 - 54. URL: https://api.semanticscholar.org/CorpusID:121808836.

[11] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805.

[12] Real Academia Española. *Corpus del Español del Siglo XXI (CORPES XXI)*. Recuperat: November 13, 2024. 2024. URL: https://www.rae.es/banco-de-datos/corpes-xxi.

[13] Hugging Face. *Hugging Face: The AI Community Building the Future*. https://huggingface.co/. Accessed: 2025-01-14. 2025.

[14] I. J. Good. "The Population Frequencies of Species and the Estimation of Population Parameters". A: *Biometrika* 40.3/4 (des. de 1953), pàg. 237 - 264. DOI: 10.2307/2333344. URL: https://doi.org/10.2307/2333344.

[15] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[16] E. T. Jaynes. *Probability Theory: The Logic of Science*. Cambridge: Cambridge University Press, 2003. ISBN: 9780521592710. URL: https://doi.org/10.1017/CBO9780511790423.

[17] Rafal Józefowicz et al. "Exploring the Limits of Language Modeling". A: *CoRR* abs/1602.02410 (2016). arXiv: 1602.02410. URL: http://arxiv.org/abs/1602.02410.

[18] Daniel Jurafsky i James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models*. 3rd. Online manuscript released August 20, 2024. 2024. URL: https://web.stanford.edu/~jurafsky/slp3/.

[19] Christopher D. Manning i Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999. ISBN: 978-0-262-13360-9.

[20] Noam Shazeer et al. "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer". A: *CoRR* abs/1701.06538 (2017). arXiv: 1701.06538. URL: http://arxiv.org/abs/1701.06538.

[21] Ilya Sutskever, Oriol Vinyals i Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". A: *CoRR* abs/1409.3215 (2014). arXiv: 1409.3215. URL: http://arxiv.org/abs/1409.3215.

[22] Ashish Vaswani et al. "Attention Is All You Need". A: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.