UNIVERSITAT POLITÈCNICA DE VALÈNCIA
ESCOLA TÈCNICA SUPERIOR D'ENGINYERIA INFORMÀTICA
DEPARTAMENT DE SISTEMES INFORMÀTICS I COMPUTACIÓ

Adaptació dels models de llenguatge per a la transcripció de vídeos de poli[Media].

Projecte Final de Carrera - Enginyeria Informàtica

Adrià Agustí Martínez Villaronga

Supervisat per:
Dr. Alfons Juan Císcar
Dr. Jesús Andrés Ferrer

25 de juliol de 2012

Als meus pares i a la meua germana.
A Fernando, Joan i Jose, amb qui he compartit
tots i cadascun d'aquests 5 anys de carrera.
A Tonga, per les ajudes lingüístiques.
A Miquel, que al final he acabat el
projecte abans que ell.
I a Jesús i a Alfons, per donar-me
l'oportunitat de treballar amb ells
i per la seua ajuda a l'hora
de fer aquest projecte

Índex

Capítol 1

Introducció

1.1 Motivació del projecte

Al llarg del darrer segle la forma de consumir la informació ha canviat de forma important. Mentre que a principis del segle XX, tot i l'incipient aparició de la cinematografia, gairebé tots els mitjans eren escrits, ara, tan sols 100 anys més tard i ficats de ple en el que s'anomena societat de la informació, aquesta apareix en tot tipus de formes i suport, mantenint-se el suport escrit (analògic o digital) com un dels més importants, però prenent gran importància també els mitjans audiovisuals. A les darreres dècades, i gràcies a internet, s'han canviat també els hàbits de consum de la informació, tenint l'usuari l'opció de triar quina informació vol i quan la vol.

En aquest context, i en l'àmbit educatiu, han aparegut iniciatives que pretenen acostar el coneixement a usuaris d'arreu del món mitjançant la distribució de classes enregistrades en vídeo. Tot i la importància d'aquestes iniciatives, presenten alguns inconvenients respecte a les fonts tradicionals de coneixement com pot ser un llibre.

El principal inconvenient, i el que més ens afecta, és el fet que gran part de la informació siga audible. Això suposa una dificultat afegida a l'hora de trobar, fent servir les tècniques clàssiques de computació, un segment o frase concreta, de forma que sovint caldrà reproduir la seqüència sencera per tal de trobar el fragment desitjat. També suposa un desavantatge considerable per a usuaris amb audició reduïda o nul·la, els quals no serien capaços d'accedir a la informació.

Una solució a aquests problemes podria ser el subtitulat dels vídeos, proporcionant un suport textual que permetria la cerca eficient de segments i facilitaria l'accés al contingut sonor del vídeo a la gent amb problemes d'audició. Tanmateix, la generació d'aquests subtítols de forma manual suposa una despesa important en temps i recursos dels quals no sempre es disposa.

La generació automàtica dels subtítols, en canvi, podria ser una solució gràcies

al seu baix cost econòmic i temporal si es compara amb la transcripció manual. Les tècniques de reconeixement de la parla, basades en les tècniques de reconeixement de formes, permetrien, a partir de la font d'àudio i d'un model prèviament entrenat, generar, de forma automàtica, subtítols per al vídeo. No obstant això, la transcripció no és perfecta i, especialment en el nostre cas on tractem amb continguts que sovint fan servir un vocabulari específic, els errors de transcripció poden dificultar la comprensió del missatge.

Un fet destacable d'aquest tipus de xarrades és que sovint van acompanyades de transparències que contenen informació textual sobre el contingut de la classe. Creiem que l'ús adequat d'aquesta informació addicional pot ajudar a millorar la qualitat de les transcripcions.

1.2 Reconeixement de formes

El reconeixement de formes o reconeixement de patrons (de l'anglès *pattern recognition*) és una disciplina englobada dins la informàtica l'objectiu de la qual és la percepció d'objectes en entorns complexos per un sistema. En aquest cas, la percepció o reconeixement d'un objecte consisteix a atorgar-li un significat semàntic a aquest mitjançant l'associació d'una etiqueta [DH73][Jel97]. El procés bàsic de reconeixement de formes consta de tres etapes (figura 1.1).

1. **Preprocés** El senyal (una imatge, una ona d'audio en cru) es processa per tal d'eliminar el possible soroll i adequar-lo als processos que se li aplicaran posteriorment.

2. **Extracció de característiques** Consisteix en obtenir una sèrie de mesures a partir del senyal preprocessat a l'etapa 1 que siguen significatives per a la tasca de classificació, la qual és l'objectiu final del reconeixement.

3. **Classificació** L'objectiu és, a partir de les dades d'entrada i en base a un model prèviament entrenat, associar una etiqueta a cada una de les mostres que es desitgen reconèixer.

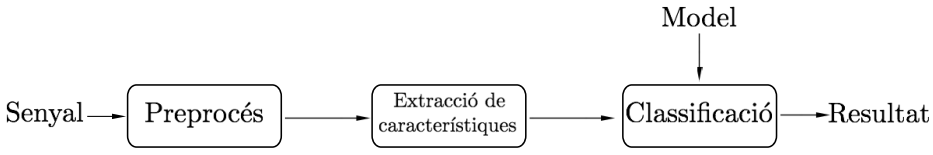

Figura 1.1: Procés de classificació en el reconeixement de formes

Com ja hem mencionat, la classificació es fa en base a un model. Per obtenir aquest model es fan servir tècniques estadístiques d'aprenentatge sobre un conjunt de dades

de les quals es coneix l'etiqueta de cada una de les mostres (conjunt d'entrenament). L'objectiu de l'entrenament és obtenir un model que s'aproxime a aquestes dades correctament etiquetades. Formalment, donada una funció desconeguda $g : X \to Y$ que associa a cada $x \in X$ una etiqueta $y \in Y$, i un conjunt de dades d'entrenament $D = \{(x_1, y_1), ..., (x_n, y_n)\}$, l'objectiu és generar una funció $h : X \to Y$ que aproxime tan precisament com siga possible la funció g. En altres paraules, volem generar una funció h que siga capaç de classificar correctament, és a dir assignar l'etiqueta $y \in Y$ correcta, al màxim nombre possible de mostres $x \in X$ d'entre un conjunt de prova.

Així, utilitzarem la regla de decisió de Bayes [**?**] per calcular l'etiqueta més probable $\hat{y}$ per a una mostr x:

$$\hat{y} = \arg\max_{y \in Y} p(y|x) \tag{1.1}$$

I aplicant la regla de Bayes podem desenvolupar l'expressió per la probabilitat a posteriori de y, $p(y|x)$, de la següent forma:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} = \frac{p(x|y)p(y)}{\sum_{y' \in Y} p(x|y')p(y')} \tag{1.2}$$

Quedant l'expressió per a l'etiqueta més probable així:

$$\hat{y} = \arg\max_{y \in Y} p(y|x) = \arg\max_{y \in Y} \frac{p(x|y)p(y)}{p(x)} \tag{1.3}$$

Com que no estem interessats en el valor de $p(y|x)$, sinó en la y de valor màxim, podem prescindir del càlcul de $p(x)$ i representar-ho com segueix:

$$\hat{y} = \arg\max_{y \in Y} p(x|y)p(y) \tag{1.4}$$

A més també podem fer servir logaritmes que ens serviran per poder mantenir la precisió treballant amb nombres molt xicotets:

$$\hat{y} = \arg\max_{y \in Y} \log(p(x|y)p(y)) = \arg\max_{y \in Y} \log p(x|y) + \log p(y) \tag{1.5}$$

És mitjançant el conjunt d'entrenament, i fent servir tècniques d'aprenentatge, que s'estimaran les probabilitats a priori de les classes, $p(y)$, i les probabilitats condicionals de x, $p(x|y)$.

1.3 Reconeixement automàtic de la parla

Dins el reconeixement de formes ens interessa especialment el reconeixement de la parla (ASR, de l'*Automatic Speech Recognition*). Aquesta és una branca especialment complexa del reconeixement de formes. Repassem com ha evolucionat el reconeixement de formes des dels seus inicis [Jel97].

Els inicis del reconeixement de veu estan als anys 50, quan a diferents laboratoris presenten diferents sistemes que porten a terme un reconeixmenet de veu molt bàsic. Estem parlant del reconeixemet de dígits aïllats monolocutor per un únic locutor, desenvolupat als Bell Labs; un reconeixedor de 10 síl·labes també monolocutor als RCA Labs; i al Lincolnl lab del MIT el reconeixedor de vocals independent del parlant.

Malgrat alguns avenços més als anys 60, com ara un sistema d'alineació dinàmica en el temps desenvolupat a la URSS o un primer intent de reconeixement de la parla contínua, no és fins als anys 70, amb l'arribada de les tècniques probabilístiques que comencen a produir-se resultats més importants com el primer sistema de reconeixement de paraules aïllades completament funcional, o alguns projectes de grans vocabularis desenvolupats per IBM.

A partir dels anys 80 es produeix una explosió de les tècniques estadístiques gràcies als models de Markov, així com també comencen a fer-se servir xarxes neuronals en tasques de reconeixement.

Les dues darreres dècades han estat plenes d'avenços, en part gràcies a l'increment de la potència dels ordinadors i els processadors, i el seu abaratiment. Als 90 apareixen els primers sistemes de dictat i de reconeixement del llenguatge natural. I ja a la primera dècada del segle XXI els sistemes de reconeixement de veu estan cada vegada més presents a l'ús diari, gràcies a la integració d'aquests als sistemes operatius tant de màquines tradicionals d'escriptori, com de dispositius mòbils.

L'objectiu del reconeixement estadístic, tal com hem explicat a la secció 1.2, és aproximar, de forma tan precisa com siga possible, la relació entre el senyal d'entrada i l'etiqueta. En el cas del reconeixement de la parla l'objectiu és, donada una seqüència d'observacions acústiques $x = x_1, \ldots, x_T$, utilitzar la regla de decisió de Bayes per decidir la seqüència de paraules $w = w_1, \ldots, w_N$ que maximitza la probabilitat a posteriori de la classe, $p(w|x)$ [Jel97]:

$$\hat{w} = \arg\max_{w} p(w|x) \tag{1.6}$$

Està demostrat que fent servir la regla de decisió de Bayes [DH73] obtenim el classificador òptim, és a dir, aquell que classificarà amb un error mínim, el problema és que, com ja hem estimat abans, la distribució de probabilitat real és desconeguda i no podem aproximar-la directament, sinó que hem de fer-ho mitjançant $P(w)$ i $P(x|w)$:

$$\hat{w} = \arg\max_{w} p(w|x) = \arg\max_{w} \frac{p(x|w)p(w)}{p(x)} = \arg\max_{w} p(x|w)p(w) \tag{1.7}$$

Aquestes dues probabilitats s'estimen fent ús de dos models prèviament entrenats: el model acústic i el model de llenguatge. La figura 1.2 mostra l'esquema d'un sistema de reconeixement de la parla, i quin paper juguen cadascun dels models en el reconeixement.

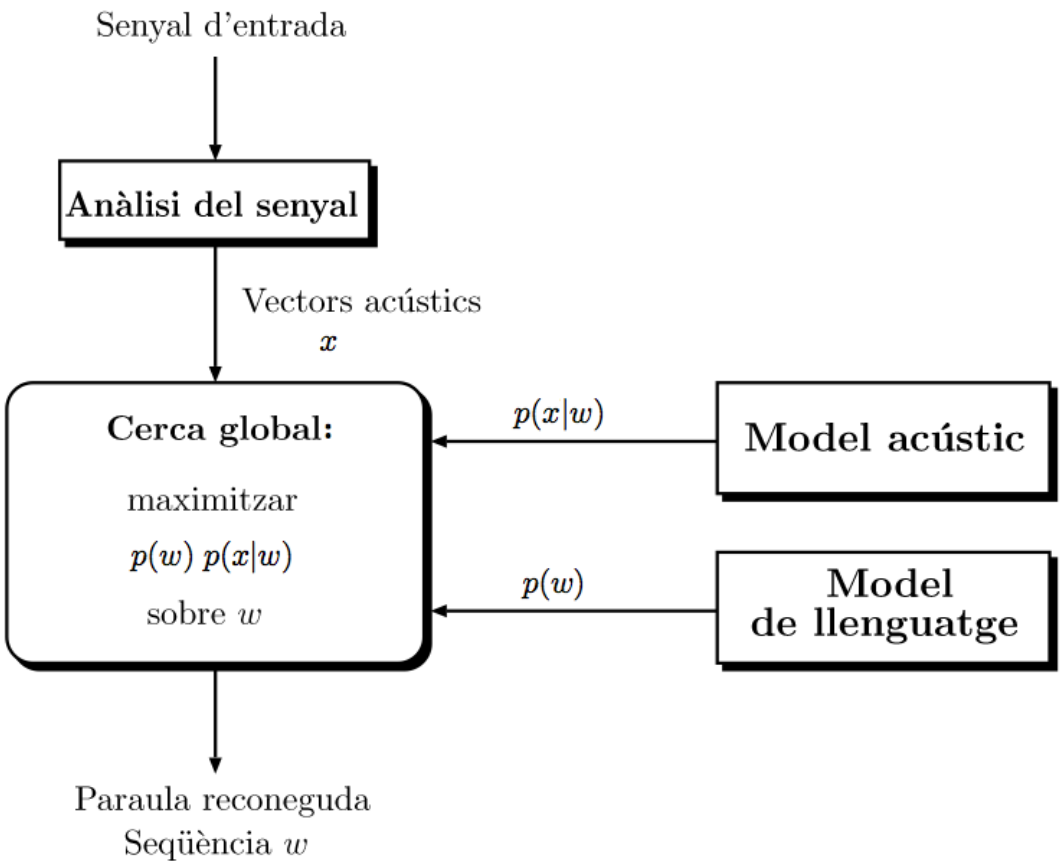

Figura 1.2: Arquitectura bàsica d'un sistema estadístic de reconeixement automàtic de la parla

1.4 Models que intervenen al reconeixement

Si recordem l'equació 1.7, per classificar una mostra necessitem conèixer $p(x|w)$ i $p(w)$. Cada una d'aquestes probabilitats ve donada per un model: el model acústic i el model de llenguatge. Aquests models s'estimaran a partir d'un conjunt de dades d'entrenament i la correcta construcció d'aquests serà clau per obtenir bons resultats al reconeixement. A la figura 1.3 es pot veure el sistema d'ASR bàsic que ja havíem presentat afegint-hi el procés d'entrenament.

El model acústic ens proporciona la probabilitat que, donada una seqüència determinada de paraules w, s'observe el vector de característiques x. Per calcular el model es fan servir els models de Markov de capa oculta (HMM, *Hidden Markov Model* en anglès), que al seu torn s'entrenen fent servir l'algorisme de Viterbi. Per entrenar-los es fan servir una sèrie de mostres d'àudio i les seves transcripcions (correctes) corresponents [GY07].

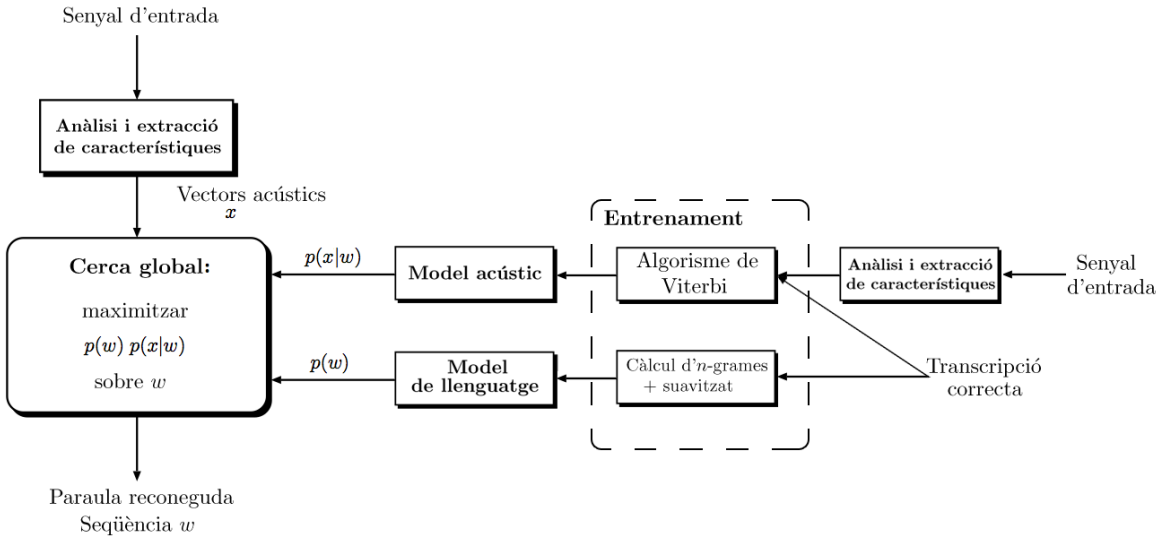

Figura 1.3: Entrenament i classificació a un sistema d'ASR

L'altre model és el model de llenguatge. El model de llenguatge proporcionarà una estimació de la probabilitat que una seqüència w es done, és a dir, associarà una probabilitat a cada seqüència possible. A diferència del model acústic, per generar aquest model és suficient disposar de transcripcions o de textos a partir dels quals poder estimar les probabilitats de cada paraula o seqüència de paraules. Els models de llenguatge que tractem en aquest projecte, els quals constitueixen el seu tema central, són els models basats en n-grames. El capítol 2 es centra en aquest tipus de models de llenguatge.

El següent exemple ens pot ajudar a fer-nos una idea de com funciona el classificador i el paper que juguen cadascun d'aquests models. Imaginem que disposem d'un sistema de reconeixement de la parla i que volem transcriure la frase següent: *ell menja a casa*. Gravem la frase amb un micròfon i després d'extreure'n les característiques obtenim x i li ho passem al classificador. El model acústic explorarà totes les possibles seqüències $w \in W$ i decidirà amb quina probabilitat podria la seqüència w presentar les característiques de x, és a dir, calcularà $p(x|w)$. Després de fer l'exploració decideix que hi ha dues seqüències $w_1 =$ *ell menja a casa* i $w_2 =$ *ell menja casa* que podrien ajustar-se amb la mateixa probabilitat a x. Un ésser humà que escolte la frase, fins i tot si el locutor no ha fet cap pausa entre les paraules *menja* i *a*, serà capaç de saber que el que probablement està dient el locutor és *ell menja a casa* i no *ell menja casa*. Aquesta és la informació que intentarà proveir el model de llenguatge. El model de llenguatge ens dirà quina frase és més comú a l'idioma, i en aquest cas sembla probable que la frase w_1 serà més comuna que la frase w_2, per tant el classificador ens oferirà com a solució *ell menja a casa*.

1.5 Avaluació dels resultats

Com a qualsevol investigació, el projecte es desenvolupa perseguint uns objectius, per tant caldrà algun tipus de mesura que ens permeta saber si s'han assolit o no, i en quin grau. La finalitat del nostre projecte era obtenir models de llenguatge que s'adaptaren millor al contingut del vídeo per millorar-ne la transcripció final, per tant caldrà triar algun tipus de mesura que ens permeta decidir si els models de llenguatge que obtenim s'ajusten millor, i especialment, en quin grau ho fan. La mesura triada és la perplexitat.

Cal saber que la perplexitat té una sèrie de característiques que fan que hi haja gent que la considere com una mala forma d'avaluar models de llenguatge. A la secció 2.4 descriurem com es calcula la perplexitat, sota quines circumstàncies la perplexitat pot ser una bona mesura i sobre quines no ho serà, i exposarem per què al nostre cas sí que és una forma adient d'avaluar els models.

Capítol 2

Modelat de llenguatges d'n-grames

Aquest capítol conforma el nucli central d'aquest projecte. En ell explicarem els conceptes relacionats amb el modelat de llenguatges i com construir-los; definirem el concepte de perplexitat i descriurem com pot ajudar-nos a l'hora d'avaluar diferents models i decidir quin s'ajusta millor als nostres objectius; i, finalment, repassarem una sèrie d'eines informàtiques per al modelat de llenguatge i que s'han fet servir en aquest projecte.

2.1 Models de llenguatge basats en n-grames

Un model de llenguatge és una distribució de probabilitat $p(w)$ sobre una cadena w que descriu la freqüència en què la cadena w apareix en un determinat domini d'interès. Els models de llenguatge no són útils únicament en tasques d'ASR, sinó en general en totes les tasques de processament del llenguatge natural com poden ser la traducció, el reconeixement d'escriptura i correcció ortogràfica [CG99].

Un n-grama és una subseqüència d'n elements d'una seqüència donada. Aquests elements poden ser lletres, paraules, frases, o altres unitats bàsiques (no necessàriament textuals), en funció de quin siga l'objectiu de la tasca. En el nostre cas cada un d'aquests grames és una paraula de la frase. La taula 2.1 mostra la descomposició en n-grames de la frase *ell menja a casa*, per a $n = 1, 2, 3, 4$.

Un model de llenguatge basat en n-grames estima la probabilitat de la frase a partir de les probabilitats dels n-grames. A un model d'n-grames la probabilitat de la frase $w = w_1 \dots w_k$ s'expressa així:

1-grames	2-grames	3-grames	4-grames
ell	ell menja	ell menja a	ell menja a casa
menja	menja a	menja a casa	
a	a casa		
casa			

Taula 2.1: Separació en n-grames de la frase *ell menja a casa*

$$p(w) = p(w_1 \dots w_k) = p(w_1)\, p(w_2|w_1)\ \dots\ p(w_k|w_1 \dots w_{k-1})$$
$$p(w) = \textstyle\prod_{i=1}^{k} p(w_i|w_1 \dots w_{i-1})$$

L'ús de la història sencera té un problema de dispersió, fent que siga molt fàcil l'aparició de probabilitats 0. La consideració d'únicament un nombre reduït de paraules anteriors reduirà en gran mesura aquesta dispersió, augmentant la densitat de les dades d'entrenament. Així, segons aquesta proposta, la probabilitat d'una frase, donat un model de trigrames es calcula així:

$$p(w) = \prod_{i=1}^{k} p(w_i|w_{i-2}w_{i-1})$$

Per exemple, amb un model de trigrames, la probabilitat de la frase *ell menja a casa* es calcularia de la següent forma:

$$p(\textit{ell menja a casa}) = p(\textit{ell}) \cdot p(\textit{menja} \mid \textit{ell}) \cdot p(\textit{a} \mid \textit{ell menja}) \cdot p(\textit{casa} \mid \textit{menja a})$$

Cal remarcar que sovint aquests models també tenen en compte l'inici i final de frase, als quals anem a denotar mitjançant $< s >$ i $< /s >$ respectivament. Això és útil perquè ajuda a poder predir el final de frase. L'exemple anterior quedaria de la següent forma si considerem els símbols d'inici i final:

$$p(\textit{ell menja a casa}) = p(\textit{ell}|\text{<s>}) \cdot p(\textit{menja} \mid \text{<s>}\ \textit{ell}) \cdot$$
$$p(\textit{a} \mid \textit{ell menja}) \cdot p(\textit{casa} \mid \textit{menja a}) \cdot p(\text{</s>} \mid \textit{a casa})$$

2.2 Construcció de models de llenguatge

Sabem que un model de llenguatge assigna una probabilitat a cada possible frase, per tant la forma més directa i senzilla de construir un model seria mitjançant un conjunt molt gran de frases en l'idioma desitjat i definir cada probabilitat com la freqüència relativa de la frase. Però per molt gran que fóra aquest conjunt d'entrenament seria impossible contemplar totes les possibles frases d'un idioma. Així, si es vol reconèixer una frase w que no s'ha vist en la fase d'entrenament, la probabilitat assignada $p(w)$

serà 0 i per tant (recordem l'equació 1.7) la probabilitat que se li assignarà a $p(w|x)$ serà també 0, per molt alta que fora la probabilitat $p(x|w)$.

Els models d'n-grames són un primer pas per intentar resoldre els problema de les probabilitats 0. Si en lloc de calcular la probabilitat d'una frase sencera la calculem a partir de les parts (n-grames) que la formen, serà molt més difícil que apareguen probabilitats 0. Anem a veure com podem obtenir la versemblança $p(w_n|w_1 \dots w_{n-1})$ que millor s'ajusta a la població que tenim:

Siga $D = \{w_1 \dots w_N\}$ un conjunt de frases sobre un vocabulari W, la probabilitat del conjunt serà:

$$p(D) = \prod_{n=1}^{N} p(w_n)$$

$$p(D) = \prod_{n=1}^{N} \prod_{l=1}^{L_n} p(w_{nl}|h_{nl})$$

On h_{nl} representa la història, és a dir, $w_{(i-k+1)l} \dots w_{i-1}l$.

Siga $\Theta = \{p(w), p(w|w'), p(w|w'w'') \quad \forall w, w', w'' \in W\}$, el conjunt de paràmetres per a un model de trigrames la versemblança del qual volem maximitzar:

$$\begin{aligned}
\hat{\Theta} &= \arg\max_{\Theta} p_{\Theta}(D) = \arg\max_{\Theta} \prod_{n=1}^{N} \prod_{l=1}^{L_n} p(w_{nl}|h_{nl}) \\
&= \arg\max_{\Theta} \log \prod_{n=1}^{N} \prod_{l=1}^{L_n} p(w_{nl}|h_{nl}) = \arg\max_{\Theta} \sum_{n=1}^{N} \sum_{l=1}^{L_n} logp(w_{nl}|h_{nl})
\end{aligned}$$

Optimitzant aquesta equació arribem:

$$p_{\mathrm{ML}}(w_n|w_1 \dots w_{n-1}) = \frac{c(w_1 \dots w_n)}{\sum_{w_n} c(w_1 \dots w_n)} = \frac{c(w_1 \dots w_n)}{c(w_1 \dots w_{n-1})} \tag{2.1}$$

on la funció $c(s)$ indica el nombre de vegades que ha aparegut la seqüència s a les dades d'entrenament. A aquesta forma d'estimar la probabilitat se la coneix com estimador de màxima versemblança (*Maximum Likelihood* en anglès).

L'estimació per màxima ver semblança ens donarà la distribució que millor s'ajusta al conjunt de dades que han servit per entrenar-la. Aquestes dades són un subconjunt de la població de possibles mostres i una distribució que s'ajuste molt bé a un subconjunt no sempre s'ajustarà bé a la població complet. A més, és evident que al subconjunt utilitzat per entrenar no apareixeran totes les paraules possibles de la població, per tant això farà que fent servir una estimació per màxima versemblança

apareguen probabilitats 0 per a n-grames que sí que estan presents a la població.

Per tant caldrà redistribuir la massa de probabilitat de tal forma que s'ajuste millor a l'univers, tot i que això supose un pitjor ajustament al conjunt d'entrenament, evitant probabilitats 0 per a paraules que no s'hagen vist durant l'entrenament. Les tècniques de suavitzat que presentem a continuació realitzen aquesta redistribució.

2.2.1 Tècniques de suavitzat

Suavitzat additiu

La forma més simple de resoldre el problema de les probabilitats 0 seria suposar que qualsevol paraula ha estat una mica més freqüent del que realment ho ha estat, per fer-ho afegim un factor δ a cada recompte, amb $0 < \delta < 1$ típicament. Per simplificar la notació, a partir d'ara expressarem $w_i \dots w_j$ com w_i^j. Així, calcularem

$$p_{\text{add}}(w_i|w_{i-n+1}^{i-1}) = \frac{\delta + c(w_{i-n+1}^{i})}{\delta|V| + \sum_{w_i} c(w_{i-n+1}^{i})} \tag{2.2}$$

on V és el vocabulari, el conjunt de totes les paraules considerades. Aquest mètode tan bàsic no sol ser suficientment bo ja que pot arribar a distorsionar molt la distribució original. Suposem les paraules w_1 i w_2 tals que $c(w_1) = 10$ i $c(w_1 w_2) = 10$. És evident que després de w_1 sempre (o, si més no, gairebé sempre) apareixerà w_2, i de fet, l'estimació per màxima versemblança així ho indicaria $p(w_2|w_1) = \frac{c(w_1 w_2)}{c(w_1)} = \frac{10}{10} = 1$. En canvi fent servir un suavitzat additiu amb un diccionari de 200 paraules i $\delta = 1$ la probabilitat nova seria $p(w_2|w_1) = \frac{c(w_1 w_2)+\delta}{\delta|V|+c(w_1)} = \frac{11}{210} = 0.052$, que no es correspon a la realitat que havíem observat.

Descompte absolut

Quan hi ha poques dades d'entrenament pot ser difícil estimar directament la probabilitat d'un n-grama $p(w_i|w_{i-n+1}^{i-1})$, no obstant, es pot fer servir la informació de l'$(n-1)$-grama corresponent, és a dir, $p(w_i|w_{i-n+2}^{i-1})$, ja que aquesta haurà estat estimada a partir de més dades. Una bona forma de combinar les probabilitats dels n-grames i dels $(n-1)$-grames és interpolant segons el model presentat per [JM80]. A [BDPDP$^+$92] donen una forma elegant de realitzar aquesta interpolació:

$$p_{\text{interp}}(w_i|w_{i-n+1}^{i-1}) = \lambda_{w_{i-n+1}^{i-1}} p_{\text{ML}}(w_i|w_{i-n+1}^{i-1}) + (1-\lambda_{w_{i-n+1}^{i-1}}) p_{\text{interp}}(w_i|w_{i-n+2}^{i-1}) \tag{2.3}$$

Partint d'aquesta idea, el suavitzat per descompte absolut [NE91] [NEK94], en lloc de multiplicar la màxima versemblança $p_{\text{ML}}(w_i|w_{i-n+1}^{i-1})$ per un factor $\lambda_{w_{i-n+1}^{i-1}}$, descompta una quantitat fixa D als recomptes diferents de 0, mentre que la part corresponent a la distribució d'ordre menor és manté igual:

$$p_{\text{abs}}(w_i|w_{i-n+1}^{i-1}) = \frac{\max\{c(w_{i-n+1}^{i}) - D, 0\}}{\sum_{w_i} c(w_{i-n+1}^{i})} + (1 - \lambda_{w_{i-n+1}^{i-1}}) p_{\text{abs}}(w_i|w_{i-n+2}^{i-1}) \tag{2.4}$$

Podem observar que, en restar D hem anat guanyant una massa de probabilitat. Aquesta és la massa que distribuirem de forma uniforme entre tots els n-grames. Aquest guany que ara li correspon a cada un dels n-grames, que és el valor de $1 - \lambda_{w_{i-n+1}^{i-1}}$, ve donat per l'expressió

$$1 - \lambda_{w_{i-n+1}^{i-1}} = \frac{D}{\sum_{w_i} c(w_{i-n+1}^{i})} N_{1+}(w_{i-n+1}^{i-1}\, \bullet)$$

La funció $N_{1+}(w_{i-n+1}^{i-1}\bullet)$ que apareix a l'equació 2.2.1 indica el nombre de paraules diferents que poden aparèixer a la posició del punt $\bullet$, és a dir, el nombre de contexts diferents que poden acompanyar a w_{i-n+1}^{i-1}. Formalment ho podem definir així:

$$N_{1+}(w_{i-n+1}^{i-1}\, \bullet) = |\{w_i : c(w_{i-n+1}^{i-1} w_i) > 0\}| \tag{2.5}$$

Kneser-Ney

Aquest suavitzat, introduït per Kneser i Ney l'any 1995 [KN95], és una extensió del descompte absolut on la distribució d'ordre menor, que s'utilitza quan la d'ordre major és nul·la o gairebé nul·la, es construeix d'una manera nova optimitzada per a aquests casos. El següent exemple pot ajudar a entendre la idea en què es basa Kneser-Ney.

En un text en valencià la paraula*york* podria ser prou comú, però amb la particularitat que la majoria de vegades anirà precedida per la paraula *nova*. Els algorismes que hem vist fins ara calculen la probabilitat dels unigrames en funció de la seua freqüència i per tant en eixos casos assignaran a $p(york)$ un valor alt. Intuïtivament podem pensar que potser és millor donar-li a l'unigrama *york* una probabilitat baixa ja que només apareix acompanyat de la paraula *nova*, cas en el qual serà el model de bigrames que modelarà correctament la probabilitat. El suavitzat de Kneser-Ney aconsegueix modelar aquesta idea intuïtiva fent la probabilitat de l'unigrama proporcional no al nombre de vegades que apareix la paraula, sinó al nombre de paraules diferents que el precedeixen.

La probabilitat $p_{\mathrm{KN}}(w_i|w_{i-n+1}^{i-1})$ segons el suavitzat de Kneser-Ney queda així:

$$p_{\mathrm{KN}}(w_i|w_{i-n+1}^{i-1}) = \frac{\max\{c(w_{i-n+1}^{i}) - D, 0\}}{\sum_{w_i} c(w_{i-n+1}^{i})} + \frac{D}{\sum_{w_i} c(w_{i-n+1}^{i})} N_{1+}(w_{i-n+1}^{i-1}\bullet) p_{\mathrm{KN}}(w_i|w_{i-n+2}^{i-1}) \tag{2.6}$$

Aquesta equació és pràcticament igual que l'equació 2.4, encara que aquí canviarà la forma de calcular el cas base de la recursió:

$$p_{\mathrm{KN}}(w_i) = \frac{N_{1+}(\bullet\, w_i)}{N_{1+}(\bullet\, \bullet)}$$

On $N_{1+}(\bullet\, w_i)$ és el nombre de contexts diferents que precedeixen a w_i, i $N_{1+}(\bullet\, \bullet)$ és la suma per a cada w_i de $N_{1+}(\bullet\, w_i)$, es a dir:

$$N_{1+}(\bullet\, w_i) = |\{w_{i-1} : c(w_{i-n}w_i) > 0\}|$$
$$N_{1+}(\bullet\, \bullet) = \sum_{w_i} N_{1+}(\bullet\, w_i)$$

Modified Kneser-Ney

Chen i Goodman proposen a [CG99] una modificació de l'algorisme de Kneser-Ney amb molt bons resultats. La proposta consisteix a no fer servir un únic valor D per als descomptes, sinó que s'utilitzaran tres paràmetres diferents D_1, D_2 i D_{3+} en funció de si l'n-grama ha aparegut una, dues o tres o més vegades així la fórmula del suavitzat queda com segueix:

$$p_{\text{KN}}(w_i|w_{i-n+1}^{i-1}) = \frac{c(w_{i-n+1}^{i}) - D(c(w_{i-n+1}^{i}))}{\sum_{w_i} c(w_{i-n+1}^{i})} + \gamma(w_{i-n+1}^{i-1}) p_{\text{KN}}(w_i|w_{i-n+2}^{i-1}) \quad (2.7)$$

on

$$D(c) = \begin{cases} 0 & \text{si } c = 0 \\ D_1 & \text{si } c = 1 \\ D_2 & \text{si } c = 2 \\ D_{3+} & \text{si } c \geq 3. \end{cases}$$

Per aconseguir que la distribució sume 1 definim $\gamma(w_{i-n+1}^{i-1})$ així:

$$\gamma(w_{i-n+1}^{i-1}) = \frac{D_1 N_1(w_{i-n+1}^{i-1}\, \bullet) + D_2 N_2(w_{i-n+1}^{i-1}\, \bullet) + D_{3+} N_{3+}(w_{i-n+1}^{i-1}\, \bullet)}{\sum_{w_i} c(w_{i-n+1}^{i})}$$

Ací, $N_k(w_{i-n+1}^{i-1}\, \bullet)$ és el nombre de contexts que apareixen exactament k vegades precedits de w_{i-n+1}^{i-1} i $N_{k+}(w_{i-n+1}^{i-1}\, \bullet)$ és el nombre de contexts que apareixen k o més vegades.

Diferències entre tècniques de *backoff* i d'interpolació

L'equació 2.3 presentava el model bàsic d'un suavitzat que combinava probabilitats d'ordre superior amb probabilitats d'ordre inferior. Aquesta forma de combinar probabilitats s'anomena interpolació. La interpolació consisteix a fer una suma ponderada entre les probabilitats d'ordre superior i d'ordre inferior. L'equació següent mostra una altra forma de representar una interpolació que s'ajusta més a la forma que tenen les equacions que hem anat veient als apartats anteriors:

$$p(w_i|w_{i-n+1}^{i-1}) = \tau(w_i|w_{i-n+1}^{i-1}) + \gamma(w_{i-n+1}^{i-1}) p(w_i|w_{i-n+2}^{i-1}) \quad (2.8)$$

En aquest cas, tant per als n-grames que no han aparegut mai com per als que sí, calculem la seua probabilitat a partir de les distribucions d'ordre superior i d'ordre

inferior. En contraposició a aquest model, el model de *backoff* només fa servir les probabilitats d'ordre inferior en cas que $c(w_{i-n+1}^{i} = 0$:

$$p(w_i|w_{i-n+1}^{i-1}) = \begin{cases} \tau(w_i|w_{i-n+1}^{i-1}) & \text{si } c(w_{i-n+1}^{i}) > 0 \\ \gamma(w_{i-n+1}^{i-1})p(w_i|w_{i-n+2}^{i-1}) & \text{si } c(w_{i-n+1}^{i}) = 0. \end{cases} \tag{2.9}$$

El model interpolat en general funciona millor [CG99] i és el que s'ha triat per als experiments. Cal no confondre la interpolació en el context del suavitzat de models de llenguatge amb la interpolació de models que explicarem tot seguit.

2.3 Interpolació de models de llenguatge

La interpolació de models és una forma de combinar models de llenguatge per crear-ne un de nou. La forma de combinar-los en la qual ens centrem en aquest projecte és la interpolació lineal. Aquesta consisteix en la suma ponderada dels diferents models a combinar.

$$\begin{aligned} p_{LM}(w_i|w_{i-n+1}^{i-1}) &= \lambda_1 p_{LM_1}(w_i|w_{i-n+1}^{i-1}) + \ldots + \lambda_n p_{LM_n}(w_i|w_{i-n+1}^{i-1}) \\ &= \sum_{k=1}^{n} \lambda_k p_{LM_k}(w_i|w_{i-n+1}^{i-1}) \end{aligned}$$

L'objectiu en interpolar dos o més models és trobar aquells valors de λ que fan que la perplexitat (veure secció 2.4) siga mínima. Per obtenir aquests valors fer servir EM, si considerem les λ com si foren les probabilitats de cadascun dels models en el model interpolat:

$$p_{LM}(w_i|w_{i-n+1}^{i-1}) = \sum_{k=1}^{n} p(LM) p_{LM_k}(w_i|w_{i-n+1}^{i-1}) \tag{2.10}$$

2.4 La perplexitat com a mesura d'avaluació d'un model de llenguatge

Per poder comparar entre diferents models de llenguatge farem servir una mesura anomenada perplexitat que es defineix de la forma següent: donat un conjunt de prova $T = \{t^{(1)}, t^{(2)}, \ldots, t^{(M)}\}$ amb M frases i N paraules en total, definim la perplexitat $PP_p(T)$ del model p sobre T com:

$$PP_p(T) = 2^{-\frac{1}{N}\sum_{m=1}^{M} \log_2 p(t^{(m)})} \tag{2.11}$$

En el cas concret dels n-grames podrem expandir la fórmula anterior:

$$PP_p(T) = 2^{-\frac{1}{N}\sum_{m=1}^{M} \log_2(\prod_{k=1}^{K^{(m)}} p(t_k^{(m)}|t_{k-n+1}^{(m)}\ldots t_{k-1}^{(m)}))} = 2^{-\frac{1}{N}\sum_{m=1}^{M}\sum_{k=1}^{K^{(m)}} \log_2 p(t_k^{(m)}|t_{k-n+1}^{(m)}\ldots t_{k-1}^{(m)})} \tag{2.12}$$

Podem entendre la perplexitat com el nombre de possibles paraules que podran aparèixer de mitjana després d'un prefix donat [AF10]. La perplexitat depèn de dos factors: l'eficiència del model i la complexitat de la tasca. Així, si comparem els models sota les mateixes circumstàncies és evident que a menys baixa perplexitat millor serà el model. Per exemple, si estem comparant diferents tècniques de suavitzat podrem considerar millor aquella que presente una menor perplexitat. No obstant això, una millora en la perplexitat no sempre indicarà una millora en el rendiment del sistema. L'exemple següent mostrarà un cas molt evident on pot baixar-se la perplexitat fins a 1, sense que això implique que el sistema és millor.

Suposem que entrenem un model de llenguatge en base a un vocabulari buit. Qualsevol paraula serà interpretada com el símbol de paraula desconeguda $< unk >$ i per tant el model de llenguatge assignarà probabilitat 1 a $p(< unk >)$, $p(< unk > | < unk >)$, $p(< unk > | < unk >< unk >)$, etc. Quan calculem la perplexitat, a causa de fer servir el vocabulari buit, interpretarem també qualsevol símbol com a $< unk >$. És evident que la probabilitat de qualsevol frase de la forma $< unk > \ldots < unk >$ serà 1 també. Com que el logaritme d'1 és zero, al final tindrem que la perplexitat és $2^0 = 1$, que és la perplexitat més baixa possible. Aquest sistema, efectivament és molt bo predient que després d'un símbol desconegut vindrà un altre símbol desconegut, però no és un bon model per al reconeixement, ja que no és capaç de predir cap paraula.

No obstant aquest problema, com ja hem dit, si comparem els models fent servir el mateix vocabulari, la perplexitat sí que serà una bona mesura per avaluar models de llenguatge.

2.5 Eines informàtiques per al modelat de llenguatges

Per treballar amb els models de llenguatge s'ha fet servir SRILM [Sto02].

SRILM és un conjunt d'eines informàtiques per a la construcció i l'aplicació de models de llenguatge estadístics per al seu ús en el reconeixement de la parla i la traducció automàtica entre d'altres. El sistema ha estat en desenvolupament des de l'any 1995 a l'SRI Speech Technology and Research Laboratory. Consta d'una sèrie de *scripts* i programes que ens serviran per construir models, interpolar-los i calcular perplexitats. Les tres eines que s'han fet servir en aquest projecte són:

`ngram-count`

Aquest programa s'encarrega de totes les tasques relacionades amb la construcció dels models, bàsicament

- Generar un model de llenguatge en format ARPA (secció 2.5.1) a partir d'un

text o d'un fitxer amb els recomptes dels n-grames. Permet especificar l'ordre del model (unigrames, bigrames, trigrames, etc.)

- Aplicar diferents tipus de suavitzat com el Kneser-Ney o el *modified* Kneser-Ney, tant en les seues versions de *backoff* com en la forma interpolada.
- Calcular els descomptes per al Kneser-Ney.
- Permet especificar un vocabulari.

`ngram`

Amb aquest programa podrem realitzar les tasques d'aplicació de models de llenguatge. L'ús que en farem nosaltres serà per calcular la perplexitat d'un text donat un model de llenguatge i calcular un model de llenguatge interpolat a partir dels models de llenguatge bàsics i dels valors de les λ.

A l'hora de calcular la perplexitat el programa per defecte mostrarà un missatge indicant entre d'altres dades el nombre de paraules del text, el nombre de frases i la perplexitat. Ara bé, una opció especialment interessant és l'opció `-debug 2` que mostrarà per a cada frase la descomposició en n-grames i la probabilitat de cada un d'ells, així com la perplexitat de cada una de les frases, a més de la perplexitat total del text. Aquesta informació serà necessària per calcular els valors de λ per a una interpolació de perplexitat mínima.

Una altra opció que ens interessa és `-skipoovs`. Per defecte, si trobem una paraula que no pertany al vocabulari calculem la seua probabilitat igual que amb les paraules que si que hi pertanyen però que no han aparegut mai a l'entrenament. L'opció `-skipoovs` ens permet saltar eixes paraules i no tindre-les en compte en el càlcul de la perplexitat.

`compute-best-mix`

Aquest *script*, donats n models de llenguatge, calcularà els valors de $\lambda_1 \dots \lambda_n$ tals que la perplexitat siga mínima. El que necessita que li passem aquest *script* és l'eixida que havíem obtingut prèviament fent servir `ngram` amb l'opció `-debug 2`.

2.5.1 El format ARPA de models de llenguatge

Els models de llenguatge amb els quals treballarem estan en format ARPA, que és el format amb el qual treballa SRILM i un dels formats utilitzats de forma més comuna. Un fitxer en format ARPA és un fitxer de text pla i és prou senzill d'entendre. Veiem un esquema d'un fitxer complet:

```
\data\
ngram 1=n1
ngram 2=n2
```

```
...
ngram N=nN

\1-grams:
p w [bow]
...

\2-grams:
p w1 w2 [bow]
...

\N-grams:
p w1 ... wN
...

\end\
```

El fitxer va introduït per la paraula clau
`data`
seguit d'n línies que indiquen el nombre d'n-grames de cada ordre. A continuació trobem els n-grames, un per línia, separats en diferents seccions segons l'ordre, on cada secció ve precedida per la paraula clau
`N-grams:`.

Per cada un dels n-grames tenim tres camps, el primer indica el logaritme en base 10 de la probabilitat condicional de l'n-grama, el segon camp són les n paraules que formen l'n-grama i el tercer camp, opcional, és el logaritme en base 10 del *backoff* per a eixe n-grama. El fitxer acaba amb la paraula clau
`end`
que indica el final del model.

Capítol 3

Descripció del corpus

Al llarg d'aquest capítol presentarem els dos corpus de dades que s'han fet servir per l'entrenament i l'experimentació al llarg del desenvolupament del projecte. En primer lloc explicarem què són i quines característiques tenen els fitxers d'n-grames de Google, i a continuació introduirem la plataforma poli[Media].

3.1 N-grames de Google

L'*Ngram Viewer* de Google books és una eina gràfica que mostra estadístiques sobre l'ús de paraules o frases (n-grames) en els 5.2 milions de llibres que Google tenia digitalitzats fins l'any 2008. La base de dades on es cerquen les paraules va ser creada l'any 2009 als Google labs i conté 500G de paraules en diferents idiomes publicades a llibres des de l'any 1500 fins al 2008. Aquesta base de dades està disponible a internet sota llicència *Creative Commons*. En aquest projecte s'han fet servir només els n-grames en castellà. Es pot trobar més informació i detalls sobre la construcció del corpus a [MSA+11].

3.1.1 Format i característiques del corpus

A causa de l'enorme quantitat d'n-grames el corpus està dividit en diferents fitxers cada un dels quals conté un fragment del corpus complet. Convé ressaltar que cada fitxer conté únicament una mida n, és dir, a un fitxer no hi haurà barrejats 1-grames, 2-grames i 3-grames. Així mateix, l'ordre màxim dels n-grames que podem trobar és 5.

Cada un dels fitxers és un fitxer de text comprimit amb *zip* que conté un n-grama per línia i cada camp està separat mitjançant l'ús de tabuladors. A continuació mostrem el format de les línies.

```
ngram TAB year TAB match_count TAB page_count TAB volume_count NEWLINE
```

El primer camp (`ngram`) és l'n-grama, amb cada una de les n paraules que el componen separades fent ús d'espais. El camp `year` indica l'any a què fa referència el recompte.

El camp `match_count` correspon al nombre total de vegades que ha aparegut l'n-grama en llibres publicats al llarg de l'any. Els camps `page_count` o `volume_count` fan referència al nombre total de pàgines i de volums on apareix l'n-grama.

A continuació es pot veure un exemple (el caràcter `__` representa la tabulació):

```
devastadores__2001__1125__1107__850
incorporen estas__1977__3__3__3
```

Als exemples anteriors podem observar que la paraula `devastadores` apareix 1125 vegades a l'any 2001, en 1107 pàgines de 850 llibres diferents. De la mateixa manera el bigrama `incorporen estas` apareix 3 vegades l'any 1977 en 3 pàgines de 3 llibres diferents.

Aquest corpus té algunes característiques que caldrà tenir en compte a l'hora de treballar amb ell i poder aplicar així els preprocessos necessaris (veure capítol 4). Algunes d'aquestes característiques que més ens afecten són:

- Diferenciació de majúscules i minúscules. El corpus de Google computa com a unigrames diferents les paraules `hola`, `Hola` i `HOLA`.
- Diferenciació de caràcters accentuats. Igual com passa amb les majúscules i minúscules, són diferents les paraules `libro` i `libró`.
- Signes de puntuació i altres símbols. Els signes de puntuació es tracten com si foren paraules diferents, de forma que si al llibre original apareixia `¿Hay`, s'haurà computat com dues paraules diferents: `¿` i `Hay`. Així i tot n'hi ha alguns que es mantenen amb la paraula ja que són necessaris per mantenir el significat de l'n-grama, per exemple en el cas de números `3.1415`. El corpus també inclou altres símbols com poden ser `$` o `#`.

3.2 poli[Media]

3.2.1 La plataforma poli[Media]

Segons el seu web [Uni12], poli[Media] és un sistema dissenyat a la UPV per a la creació de continguts multimèdia com a suport a la docència presencial, que comprèn des de la preparació del material docent fins a la distribució als destinataris a través de diferents mitjans (TV, Internet, CD, etc.).
Les característiques bàsiques del servei són:

- poli[Media] és un sistema de producció de materials educatius de qualitat.
- És un recurs integrat amb totes les eines de poli[formaT].
- És molt adient com a suport i complement a l'ensenyament presencial.
- L'autor és el propietari intel·lectual de l'obra.

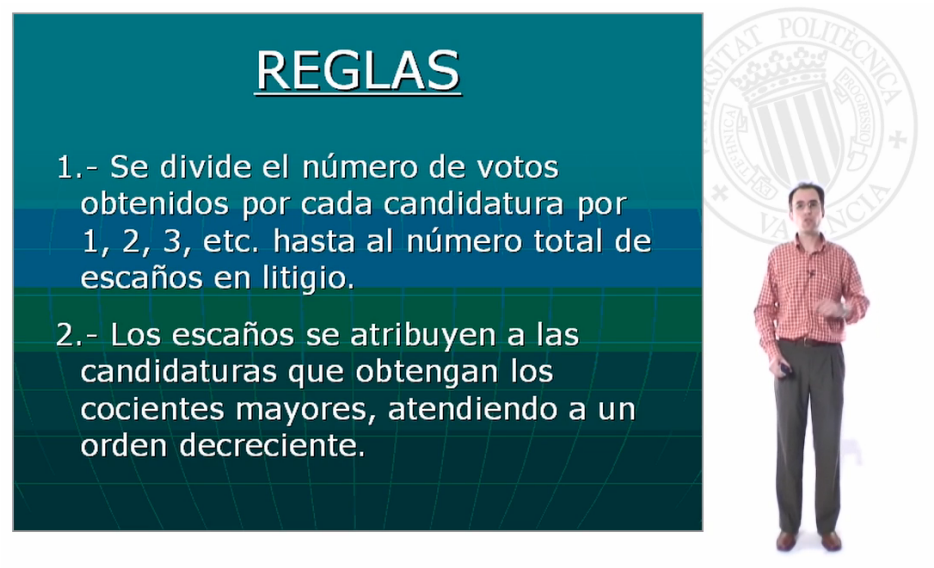

Figura 3.1: Captura de pantalla d'un vídeo de poli[Media]

- Sistema completament innovador i únic, disponible només en la UPV.
- Disponibilitat dels millors instruments, materials i tècnics al servei del professorat.
- Porta associat un pla d'incentius econòmics.
- Fàcil: no requerix coneixements audiovisuals o tècnics.

A data de 12 de juliol de 2012, poli[Media] té a disposició dels usuaris 8139 vídeos amb una durada total superior a les 1420 hores.

Tots els vídeos de poli[Media] estan enregistrats a un estudi de gravació sota unes condicions molt controlades. S'enregistra el professor sobre un fons blanc i simultàniament s'enregistra també la pantalla de l'ordinador (generalment transparències), després es munta de forma que les transparències queden com a fons del vídeo i es superposa el professor. Una captura del resultat final es pot veure a la figura 3.1.

3.2.2 El corpus poli[Media]

Per a aquest projecte s'han fet servir transcripcions i transparències d'un subconjunt de vídeos de poli[Media]. En total el corpus conté 739 vídeos de diferents matèries i professors dels quals 690 es faran servir per entrenar el model (*train*), 26 funcionaran com a conjunt de desenvolupament (*development*) i els 23 restants serviran per validar el model (*test*). També disposem de les transparències corresponents als conjunts de desenvolupament i de *test*. La taula 3.1 mostra amb més claredat la quantitat de vídeos utilitzats i quin percentatge del total representen; mentre que a la taula 3.2

	train	*dev*	*test*	Corpus	poli[Media]
Nre. de vídeos	690	26	23	739	8139
Percentatge sobre el total (%)	8.48	0.32	0.28	9.08	100

Taula 3.1: El corpus utilitzat en relació a poli[Media] complet

	train	*dev*	*test*
Nre. de vídeos	690	26	23
Nre. de frases	$41.5K$	$1.4K$	$1.1K$
Nre. de paraules	$96.8K$	$34K$	$28.7K$
Nre. de paraules diferents (vocabulari)	$28K$	$4.5K$	$4K$

Taula 3.2: Descripció numèrica del corpus poli[Media]

podem veure informació més detallada sobre els vídeos.

Cal destacar que tots els vídeos i transparències del corpus han estat transcrits manualment, la qual cosa garanteix una transcripció perfecta.

3.2.3 Format de les transcripcions

Tot i que el procés de transcripció no forma part d'aquest projecte, convé saber quins criteris s'han seguit a l'hora de transcriure, així com el format que s'ha fet servir per emmagatzemar aquestes transcripcions:

1. S'identificarà el professor que per a cada vídeo, així com el seu sexe.

2. Se segmentarà l'àudio en frases, transcrivint cada una d'elles.

3. Quan es produïsca una disfluència, s'anotara fent servir la següent notació estàndard global: `/so pronunciat/paraula correcta/`

4. Quan es produïsca un silenci llarg es crearà un segment la transcripció del qual serà `[sonido de fondo]`.

5. Quan es produïsca un silenci curt s'afegirà la notació `/SF//`.

Així doncs, el resultat d'aquesta transcripció és un fitxer XML, amb les seues capçaleres corresponents, on cada frase és transcrita segons els criteris anteriors i va precedida d'una marca de temps que indica l'instant d'inici de la frase. A continuació mostrem un fragment del cos d'una de les transcripcions:

```
<Sync time="11.675"/>
/Toces/Entonces/ este título tan dramático vamos a ver sobre todo /e//
la pérdida de profundidad por una parte, /SF// y la pérdida de sonido
ambiente que ocurre con el doblaje.
<Sync time="23.621"/>
[sonido de fondo]
<Sync time="24.855"/>
Entonces vamos /a// a ver /e// los cambios que acarrea /la// /las obra//
en las obras audiovisuales el doblar /SF// entonces vamos a ver varios
ejemplos /SF// para entender qué elementos se modifican y entonces
podremos /e// desarrollar digamos unas conclusiones.
<Sync time="44.83"/>
[sonido de fondo]
```

Per a les transparències el format és similar. En aquest cas no hi ha silencis ni disfluències possibles, per tant no hi haurà cap d'aquestes marques al fitxer. Tampoc és necessària en aquest cas la informació sobre el locutor. El que sí que trobem són marques de temps que indiquen l'instant en què comença la transparència. Cada frase va entre uns marcadors que indiquen el número de frase a la transparència. El fragment següent correspon una de les transparències del conjunt de *dev*.

```
<Trasp>
<Sync time="76"/>
<1>Título X de la Constitución</1>
<2>"De la reforma constitucional"</2>
<3>- Iniciativa de reforma (art. 166 remite a art. 87)</3>
<4>El Gobirno, el Congreso y el Senado y las Asambleas legislativas de
Comunidades Autónomas.</4>
<5>- Procedimientos de reforma.</5>
<6>Dos procedimientos:</6>
<7>1. Reforma total o parcial que afecte al Título Preliminar, al
Capítulo II, Sección 1a del Título I, o al TítuloII, se seguirá un
procedimiento superagravado (ar. 168).</7>
<8>2. Reforma parcial (art. 167)</8>
<Trasp>
<Sync time="173"/>
<1>Procedimiento especial agravado (art. 168)</1>
<2>- Mayoría 2/3 de cada Cámara.</2>
<3>- Disolución de las Cámaras.</3>
<4>- Ratificar decisión nuevas Cámaras y aprobación 2/3 de cada
Cámara.</4>
<5>- Reforma sometida a referéndum para ratificación.</5>
```

Capítol 4

Experimentació

El present capítol tractarà d'explicar els experiments que s'han portat a terme. Podem distingir dos blocs d'experimentació. Un bloc que ens serveix de base per saber quin és l'estat del qual partim, i un segon bloc on es porten a terme els experiments d'adaptació. Aquesta adaptació la farem interpolant diferents models, segons la tècnica explicada a la secció 2.3. Els models que obtindrem al bloc 1 i dels quals calcularem la perplexitat són:

- Model de llenguatge de Google
- Model de llenguatge de poli[Media]
- Model interpolat Google + poli[Media]

El segon bloc ja tindrà en compte les transparències de les xarrades i també consta de tres experiments:

- Model interpolat poli[Media] + transparències
- Model interpolat Google + poli[Media] + transparències
- Model interpolat Google + poli[Media] + transparències + transparències sincronitzades

Al llarg del capítol explicarem com s'han construït els diferents models i quins resultats hem obtingut amb cada un d'ells, així com el preprocés que se'ls ha aplicat a les dades abans de començar a treballar amb elles. Cal remarcar també que en la construcció dels models, si no especifiquem el contrari, es farà servir el suavitzat *modified* Kneser-Ney en la seua versió interpolada (veure secció 2.2.1).

4.1 Preprocés de les dades

Al capítol 3 explicàvem el format en què estaven les dades. Hem vist, per exemple, que Google emmagatzema els seus n-grames separats per anys, o que en el cas del

corpus poli[Media] s'identifica el locutor per saber si és home o dona. Aquest tipus d'informació, que pot ser útil o necessària per a altres tasques no és interessant per al modelat de llenguatges, on l'única cosa que necessitem és una llista de paraules i les seues freqüències, per tant serà desitjable aplicar un preprocés per tal de mantenir només aquelles dades que ens siguen d'interès, per facilitar així el treball amb els fitxers.

4.1.1 Preprocés dels fitxers de Google

Si recordem els fitxers de google (secció 3.1) teníem les aparicions de cada n-grama separades per anys i per a cada any, a més del nombre de vegades que apareix l'n-grama també s'emmagatzemava el nombre de llibres i de pàgines on apareixia. L'única dada que ens interessa és la freqüència, per tant la resta de camps els desestimarem. De cara al reconeixement de veu, també ens és igual si la paraula *hola* ha aparegut com `hola`, `Hola` o `HOLA`. És la mateixa paraula i es pronuncia igual, per tant també caldrà aplicar-li un filtrat als n-grames per passar-los a minúscules. Els símbols de puntuació tampoc no es pronuncien i per tant tampoc no els tindrem en compte. També hem decidit filtrar els accents, de forma que la paraula *comprensión* apareixerà com `comprension`. Una altra decisió que s'ha pres ha sigut ometre els números, ja que aquests no es pronuncien.

Una vegada fet tot això, tindrem diverses entrades per a una mateixa paraula i caldrà combinar-les. Suposem l'exemple següent:

```
él 2000 5 3 4
él 2001 7 2 2
Él 2000 2 2 2
ÉL 2000 3 2 1
ÉL 2001 4 2 3
el 2000 13 7 4
el 2001 19 5 12
El 2000 7 3 5
El 2001 5 4 5
EL 2000 2 1 2
EL 2001 3 2 2
```

Una vegada preprocessat obtindrem el següent:

```
el 5
el 7
el 2
el 3
el 4
el 13
el 19
el 7
```

```
el 5
el 2
el 3
```

I totes aquestes entrades d'`el` s'hauran de combinar en una única, ja que assumim que totes es pronuncien igual:

```
el 70
```

Per últim caldrà filtrar per vocabulari, de forma que aquells n-grames que continguen paraules de fora del vocabulari no es tindran en compte. Al final, per motius que explicarem més endavant, caldrà guardar dues versions dels fitxers de recomptes, una amb filtratge per vocabulari i una altra sense filtrar.

4.1.2 Preprocessat de poli[Media]

En el cas de poli[Media] recordem que el nostre corpus està format per fitxers XML amb les seues etiquetes corresponents, les quals caldrà eliminar. Un cas especial d'aquestes etiquetes són les marques de temps (`<Sync time="XXX"/>`), que si que caldrà tenir-les en compte quan es faça la separació per transparències.

A més d'eliminar les etiquetes d'XML també haurem d'eliminar les marques de silenci (`[sonido de fondo]` i `/SF//`) així com les marques que indiquen una disfluència `/so pronunciat/paraula correcta/`, quedant-nos en aquest cas amb l'expressió de la dreta. Aquesta part només serà necessària en les transcripcions de l'àudio; per a les transparències no serà necessària.

El següent pas serà eliminar signes de puntuació i accents i passar-ho tot a minúscules, seguint els mateixos criteris que havíem seguit amb Google, de forma que els dos corpus siguen compatibles.

Per últim caldrà separar les transcripcions i les transparències de forma que cada un dels fitxers resultants corresponga a una única diapositiva. Això és necessari per a la tasca de Google + poli[Media] + transparències + transparències sincronitzades.

4.2 Construcció del vocabulari

El nostre vocabulari estarà format per:

- Els 50000 unigrames més freqüents de Google després del preprocés.
- Totes les paraules que apareguen al conjunt d'entrenament de poli[Media]
- Les paraules que apareixen a les transparències

El vocabulari final, unint aquests tres conjunts de paraules, consta de 62792 paraules.

4.3 Experiments bàsics

Aquests són els models de partida.

4.3.1 Model de llenguatge de Google

Per calcular el model de Google ho farem a partir dels recomptes que hem generat en el preprocés. Primer hem de calcular els descomptes del Kneser-Ney. Els descomptes es calcularan en base al corpus de Google complet, aquell que no havíem filtrat per vocabulari. Amb els descomptes calculats ja podem calcular el ML, ara sí, a partir dels recomptes filtrats de forma que només contemplarem n-grames on totes les paraules formen part del vocabulari.

Amb el model construït calcularem la perplexitat del model sobre *dev* i sobre *test*. En tots els casos calcularem la perplexitat fent servir l'opció *skipoovs* i també sense fer-la servir, per poder comparar com es comporten els models quan no es tenen en compte les paraules que no pertanyen al vocabulari.

La taula 4.1 mostra els resultats obtinguts.

	Perp. sense l'opció *skipoovs*	Perp. amb l'opció *skipoovs*
Perp. sobre *dev*	1295.62	1019.11
Perp. sobre *test*	1907.54	1554.07

Taula 4.1: Resultats de calcular la perplexitat per al ML de Google sobre *dev* i *test*

4.3.2 Model de llenguatge de poli[Media]

Calculem els descomptes a partir del text d'entrenament i construïm el model de llenguatge. Després calcularem la perplexitat sobre *dev* i sobre *test*. La taula 4.2 mostra els resultats.

	Perp. sense l'opció *skipoovs*	Perp. amb l'opció *skipoovs*
Perp. sobre *dev*	290.79	291.98
Perp. sobre *test*	320.14	324.89

Taula 4.2: Resultats de calcular la perplexitat per al ML de poli[Media] sobre *dev* i *test*

4.3.3 Model interpolat de Google + poli[Media]

Els models de llenguatge de Google i de poli[Media] són els que ja havíem calculat a les seccions 4.3.1 i 4.3.2 respectivament. Una vegada s'han obtingut els models, cal

generar els fitxers de *debug* (secció 2.5) sobre els vídeos de *dev*, i a partir dels resultats obtenir els valors λ_G i λ_p que minimitzen la perplexitat sobre el conjunt de *dev*. Per obtenir-los es fa servir la interpolació lineal vista a la secció 2.3. Els valors obtinguts són els que es presenten a la taula 4.3.

	λ sense skipoovs	λ amb skipoovs
λ_G	0.384	0.399
λ_p	0.616	0.601

Taula 4.3: Valors de λ_G i λ_p que minimitzen la perplexitat sobre *dev*

Amb els valors de λ_G i λ_p calculats generarem el nou model $p_{ML_{G+p}}(w_i|w_{i-n+1}^{i-1}) = \lambda_G p_{ML_G}(w_i|w_{i-n+1}^{i-1}) + \lambda_p p_{ML_p}(w_i|w_{i-n+1}^{i-1})$ i fem el càlcul de la perplexitat sobre els conjunts de *dev* i de *test*, els resultats del qual es troben a la taula 4.4.

	Perp. sense l'opció *skipoovs*	Perp. amb l'opció *skipoovs*
Perp. sobre *dev*	170.20	166.94
Perp. sobre *test*	204.92	203.30

Taula 4.4: Resultats de calcular la perplexitat per al ML Google + poli[Media] sobre *dev* i *test*

Amb aquests tres models donem per acabats els experiments bàsics. El model de Google + poli[Media] serà el que farem servir com a base amb el qual compararem els resultats que obtinguem amb els nostres models adaptats.

4.4 Model interpolat poli[Media] + transparències

La primera prova d'adaptació que farem serà utilitzar per a cada vídeo el model de llenguatge de poli[Media] interpolat amb el model de llenguatge de les transparències corresponents.

El model de llenguatge de poli[Media] no varia i serà el mateix de la secció 4.3.2. Per a les transparències, cal generar un ML de llenguatge diferent per a cada un dels vídeos. Atesa la poca extensió de cada un dels vídeos pot ser que la quantitat de dades siga insuficient per a poder calcular els descomptes de *modified* Kneser-Ney. Si es dóna el cas, s'intentarà fer servir el Kneser-Ney original i, si això no fos, possible es farà servir un descompte constant de 0.8.

El fet que els fragments siguen tan menuts farà que potser un model de trigrames funcione pitjor que un model d'ordre menor. És per això que es generaran models de llenguatges per a unigrames, bigrames i trigrames per, posteriorment, saber quin

model convé més quedar-se.

Amb tots els models ja entrenats caldrà obtenir els fitxers de *debug*. Per a poli[Media] serveix el que ja teníem calculat de la secció 4.3.2. En el cas dels vídeos caldrà aplicar cada ML al vídeo corresponent. Això ens generarà un fitxer de *debug* per a cada vídeo però, per poder fer la interpolació, ens cal tenir el fitxer de *debug* del text complet, per tant caldrà combinar tots els fitxers generats per obtenir un nou fitxer amb la perplexitat del ML dels vídeos per al text complet. Els fitxers de *debug* també s'hauran de generar per als models d'unigrames, bigrames i trigrames.

Amb els fitxers de *debug* dels dos models ja construïts només cal calcular les lambdes de perplexitat mínima. Una vegada sapiguem quina és aquesta perplexitat podrem decidir quin model funciona millor per als vídeos, si unigrames, bigrames o trigrames. Els valors de les λ i les perplexitats obtingudes en cada cas es troben a les taules 4.5 i 4.6.

Per al càlcul de les perplexitats en *test* caldrà calcular-les individualment per a cada vídeo i després obtenir la perplexitat total, de la mateixa forma que havíem fet per obtenir la de *dev*.

	λ 1-grames	λ 2-grames	λ 3-grames
λ_p	0.761	0.664	0.648
λ_t	0.239	0.336	0.352
λ_p amb *skipoovs*	0.753	0.665	0.655
λ_t amb *skipoovs*	0.247	0.335	0.345

Taula 4.5: Valors de λ_p i λ_t que minimitzen la perplexitat sobre *dev*

	1-grames	2-grames	3-grames
Perp. sobre *dev*	190.35	130.71	117.93
Perp. sobre *dev* amb *skipoovs*	200.85	133.97	117.18
Perp. sobre *test*	-	-	127.70
Perp. sobre *test* amb *skipoovs*	-	-	123.95

Taula 4.6: Resultats de calcular la perplexitat per al ML poli[Media] + transparències sobre *dev* i *test*

4.5 Model interpolat Google + poli[Media] + transparències

El desenvolupament d'aquest experiment és igual que l'anterior però afegint-hi el model de Google. Els valors de λ obtinguts i les perplexitats es mostren a les taules

4.7 i 4.8.

	λ 1-grames	λ 2-grames	λ 3-grames
λ_G	0.354	0.293	0.278
λ_p	0.488	0.437	0.429
λ_t	0.158	0.270	0.293
λ_G amb *skipoovs*	0.353	0.296	0.283
λ_p amb *skipoovs*	0.480	0.435	0.432
λ_t amb *skipoovs*	0.166	0.269	0.285

Taula 4.7: Valors de λ_G, λ_p i λ_t que minimitzen la perplexitat sobre *dev*

	1-grames	2-grames	3-grames
Perp. sobre *dev*	135.81	101.73	92.85
Perp. sobre *dev* amb *skipoovs*	143.21	103.73	91.61
Perp. sobre *test*	-	-	105.07
Perp. sobre *test* amb *skipoovs*	-	-	101.06

Taula 4.8: Resultats de calcular la perplexitat per al ML Google + poli[Media] + transparències sobre *dev* i *test*

4.6 Model interpolat Google + poli[Media] + transparències + transparències sincronitzades

Aquest és l'experiment final amb el qual esperem obtenir la mínima perplexitat. El procediment serà molt similar als dos anteriors, però caldrà dividir el corpus a nivell de transparència, enlloc d'a nivell de vídeo. Igual que passava en el cas dels vídeos, i tenint en compte que ara cada model es construirà a partir d'una unitat d'extensió molt més menuda com és una transparència, també caldrà construir models de diferents ordres per a cada una de les transparències i triar aquell que tinga un millor comportament. Pel mateix motiu també pot passar que a l'hora de construir el model no hi haja informació per calcular els descomptes de Kneser-Ney i de *modified* Kneser-Ney, i en aquest cas caldria fer servir un descompte constant. Com que a l'experiment anterior havíem vist que els millors resultats s'obtenien construint models de trigrames per als vídeos, per aquest experiment s'han fet servir directament aquests models i s'ha variat només l'ordre dels models sincronitzats a nivell de transparència.

4.7 Resum dels resultats

Fets tots els experiments, a la taula 4.11 podem veure com, a mesura que el model s'adapta millor al vídeo, les perplexitats es van reduint. A la figura 4.1 es pot veure, de forma més gràfica, aquesta evolució per als models sense *skipoovs*.

	λ 1-grames	λ 2-grames	λ 3-grames
λ_G	0.272	0.269	0.269
λ_p	0.416	0.408	0.408
λ_t	0.266	0.220	0.197
λ_s	0.046	0.103	0.126
λ_G amb *skipoovs*	0.272	0.272	0.275
λ_p amb *skipoovs*	0.407	0.404	0.409
λ_t amb *skipoovs*	0.245	0.202	0.183
λ_s amb *skipoovs*	0.076	0.122	0.133

Taula 4.9: Valors de λ_G, λ_p, λ_t i λ_s que minimitzen la perplexitat sobre *dev*

	1-grames	2-grames	3-grames
Perp. sobre *dev*	89.45	85.52	84.10
Perp. sobre *dev* amb *skipoovs*	100.17	90.86	85.11
Perp. sobre *test*	-	-	105.98
Perp. sobre *test* amb *skipoovs*	-	-	101.89

Taula 4.10: Resultats de calcular la perplexitat per al ML Google + poli[Media] + transparències + sincronització sobre *dev* i *test*

	G	p	G + p	p + t	G + p + t	G + p + t + s
Sense *skipoovs*	1907.54	320.14	204.92	117.93	105.07	105.98
Amb *skipoovs*	1554.07	324.89	203.30	117.18	101.06	101.89

Taula 4.11: Evolució de les perplexitats segons el ML utilitzat

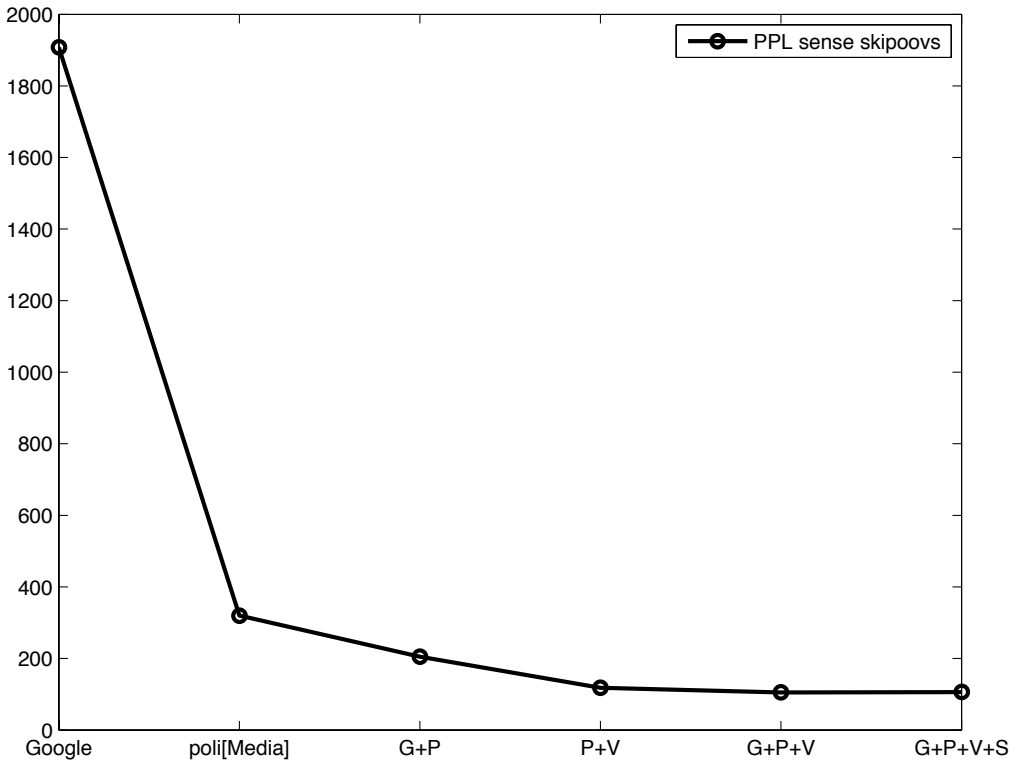

Figura 4.1: Evolució de la perplexitat amb els diferents models provats

Capítol 5

Conclusions

En aquest projecte pretenïem millorar la qualitat de les transcripcions automàtiques en el context dels vídeos amb xarrades docents. Per això volíem aprofitar el fet que aquest tipus de xarrades solen anar acompanyades de transparències.

En el context del reconeixement automàtic de la parla hi ha dos models que juguen un paper fonamental, el model acústic i el model de llenguatge. El nostre objectiu era crear models de llenguatge específics per a cada vídeo combinant, mitjançant la tècnica de la interpolació, models de llenguatge més general amb models extrets a partir de les transparències. Per avaluar els models obtinguts farem servir la perplexitat, de manera que, com més baix siga el valor serà millor el resultat.

Per treballar amb models de llenguatge s'han fet servir les eines de modelatge de llenguatges SRILM, que permeten generar models de llenguatge a partir d'un text, d'un fitxer amb recomptes o mitjançant la interpolació de dos o més models i aplicar diferents tècniques de suavitzat. Les eines d'SRILM també permeten calcular les perplexitats dels models sobre un text i calcular els valors dels pesos per a la interpolació que minimitzen la perplexitat.

S'han fet servir dos corpus de dades: d'una banda, els fitxers d'n-grames de Google, extrets dels llibres de Google books des de 1500 fins a 2008, s'han utilitzat per entrenar un model general; d'altra, banda el corpus poli[Media] format per 739 vídeos d'aquesta plataforma, així com les transparències de 49 d'ells corresponents als conjunts de *dev* i de *test*.

Sobre aquests conjunts de dades s'han portat a terme una sèrie d'experiments consistents a calcular la perplexitat que tenen diferents models de llenguatge sobre el conjunt de *test*. Els primers experiments, sense adaptació específica als vídeos, corresponien als models de llenguatge de Google, de poli[Media] i a la interpolació de Google i poli[Media], aquest darrer amb una perplexitat de 204 sobre *test*. A partir d'aquest punt s'han realitzat tres experiments més:

- Interpolant poli[Medi] amb un model adaptat a nivell de vídeo.

- Interpolant Google, poli[Media] i el model a nivell de vídeo.
- Interpolant Google, poli[Media], el model adaptat a nivell de vídeo i un model adaptat a nivell de transparències

obtenint perplexitats de 117, 105 i 105 respectivament, de manera que es pot observar una evident millora.

5.1 Treball futur

En vista dels prometedors resultats obtinguts a nivell de perplexitat, s'obren dues línies d'investigació fonamentals de cara al futur.

D'una banda caldria provar els models obtinguts sobre vídeos reals. És evident que els resultats que hem obtingut fan pensar que la tasca de reconeixement millorarà notablement de fer servir un model no adaptat a fer-ne servir un d'adaptat, fins i tot el més senzill dels tres que hem vist, però caldria provar-los sobre vídeos reals i mesurar, fent servir per exemple el *WER* el nivell de millora obtingut. També es podrien provar els models generats en tasques de traducció, ja que els models de llenguatge són una part important d'aquestes tasques. Creiem que aquesta forma de generar models de llenguatge adaptats pot portar a millorar les traduccions automàtiques d'aquestes xarrades.

L'altra línia a seguir és continuar la investigació sobre el model actual per tractar de millorar encara més els resultats obtinguts. Tot seguit es mostren unes d'aquestes possibles millores:

- Investigar com afecta el canvi dels diferents paràmetres, com poden ser la mida del vocabulari, el valor del descompte constant o l'ordre màxim dels n-grames utilitzats.
- Fer servir la interpolació loglineal per màxima entropia enlloc de la interpolació lineal que hem fet servir.
- Construir els models fent ús de tècniques basades en xarxes neuronals.

Així doncs, amb els bons resultats obtinguts i les línies de treball obertes es presenten moltes oportunitats per seguir treballant en un camp de gran utilitat i projecció, i amb perspectives d'aportar millores significatives a la transcripció de vídeos docents.

Bibliografia

[AF10] Jesús Andrés Ferrer. *Statistical approaches for natural language modelling and monotone statistical machine translation.* PhD thesis, Universitat Politècnica de València, 2010. Advisors: A. Juan and F. Casacuberta.

[BDPDP+92] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. C. Lai, and R. L. Mercer. An estimate of an upper bound for the entropy of english. *Computational Linguistics*, 18:31–40, 1992.

[CG99] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393, 1999.

[DH73] R. O. Duda and P. E. Hart. *Pattern classification and scene analysis.* John Wiley, New York, 1973.

[GY07] Mark Gales and Steve Young. The application of hidden markov models in speech recognition. *Foundations and Trends ® in Signal Processing*, 1(3):195–304, 2007.

[Jel97] Frederick Jelinek. *Statistical methods for speech recognition.* MIT Press, Cambridge, MA, USA, 1997.

[JM80] F. Jelinek and R. L. Mercer. Interpolated estimation of markov source parameters from sparse data. *Proceedings of the Workshop on Pattern Recognition in Practice*, pages 381–397, may 1980.

[KN95] R. Kneser and H. Ney. Improved backing-off for m-gram language modelling. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 1:181–184, 1995.

[MSA+11] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. Quantitative analysis of culture using millions of digitized books. *Science*, 331(6014):176–182, 2011.

[NE91] H. Ney and U. Essen. On smoothing techniques for bigram-based natural language modelling. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 2:825–829, 1991.

[NEK94] H. Ney, U. Essen, and R. Kneser. On structuring probabillistic dependences in stochastic language modeling. *Computer Speech and Language*, 3:1–38, 1994.

[Sto02] A. Stolcke. Srilm - an extensible language modeling toolkit. Proc. Intl. Conf. Spoken Language Processing, Denver, Colorado, September 2002.

[Uni12] Universitat Politècnica de València. poli[media]. https://polimedia.upv.es/, 2012.

Índex de figures

Índex de taules