Dialbot: muturretik muturrerako dialogo sistema

Julen Etxaniz
University of the Basque Country
jetxaniz007@ikasle.ehu.eus

Aitor Zubillaga
University of the Basque Country
azubillaga012@ikasle.ehu.eus

Laburpena

Proiektu honetan ikasketa sakonean oinarritutako muturretik muturrerako solasaldi sistema bat garatu da (Bahdanau et al., 2014) lanean oinarritua eta filmetako azpitituluak erabiliz (Lison and Tiedemann, 2016).

1 Sarrera

Proiektu honetan ikasketa sakonean oinarritutako muturretik muturrerako solasaldi sistema bat garatu da (Bahdanau et al., 2014) lanean oinarritua eta filmetako azpitituluak erabiliz (Lison and Tiedemann, 2016). Honetarako, dialogoa itzulpen ataza bat bezala proposatu da, hurrengo adibidean ikus daitekeen bezala:

- Itzulpen automatikoa:
  - Sistemaren sarrera - esaldia jatorri hizkuntza batean: "Egun on guztioi."
  - Sistemaren irteera - esaldia helburu hizkuntzan: "Buenos días a todos."
- Dialogoa:
  - Sistemaren sarrera - dialogoko partaide baten esaldia: "Egun on guztioi."
  - Sistemaren irteera - sarrerako esaldiari erantzuna: "Baita zuri ere."

2 Helburuak

Proiektu honen helburuak honakoak dira:

1. Ingeleserako entrenatua izan den muturretik muturrerako solasaldi sistemaren eredua deskargatu eta probatzea. Honetaz gain, sistemaren arkitektura eta entrenamendurako erabilitako datuak aztertzea.
2. Euskarazko filmen azpitituluekin eredu berri bat entrenatzea.
3. Telegrameko bot bat sortzea, aurretik aipatutako bi ereduak integratzen dituena.
4. Aurreko txandako galdera kontuan erabilita dialogoaren testuingurua kontuan hartzen duen sistema berri bat garatzea.

3 Erlazionatutako lanak

Proiektu honetan ikasketa sakonean oinarritutako muturretik muturrerako solasaldi sistema bat garatu da (Bahdanau et al., 2014) lanean oinarritua eta filmetako azpitituluak erabiliz (Lison and Tiedemann, 2016). Hiztegia sortzeko Byte Pair Encoding (BPE) algoritmoa erabili dugu (Sennrich et al., 2015).

Oinarri moduan irakasleak emandako kodea erabili dugu. Bertan ingeleserako entrenatutako eredua dago. Gainera, ingeleseko sistema entrenatzeko erabili diren datuak eta kodea daude.

Gure kodea hedatzeko baliabide gehiago erabili ditugu. Alde batetik, Pytorch-eko tutorialak erabili dira. Bestetik, Ben Trevett-en tutorialak ere erabilgarriak izan dira.

4 Datuak

Esan bezala, filmetako azpitituluak erabiliko ditugu gure ereduak entrenatzeko. Hurrengo helbidean hizkuntza askotako azpititulu fitxategiak dituzu: http://opus.nlpl.eu/OpenSubtitles-v2018.php. Ereduak entrenatzeko erabili den euskarazko fitxategia hemen aurki daiteke. Fitxategi honek milioi bat lerro inguru ditu. Lerro bakoitzean dialogoko partaide batek esaten duena dago.

Datuak filmetako azpitituluak izateak ereduak ikasi dezakeena baldintzatzen du. Horregatik, ingeleseko eredua entrenatzeko beste datu batzuk erabili dira, pertsonen arteko elkarrizketak. Euskararako aukera hori egongo balitz seguruenik hobea izango litzeteke.

5 Sistema

5.1 Aurreprozesamendua

Sarea entrenatu ahal izateko euskarazko dataseta ingelesekoaren formatu berdinean jarri dugu.

Hasteko, `eu.txt` fitxategiko 500.000 lerro irakurriko ditugu. Ondoren, lerro bakoitza garbituko dugu, letrak eta `.?!` puntuazio ikurrak bakarrik utziz. Gero, testua tokenizatuko dugu, esaldiko tokenak espazio batekin banatuz. Jarraian, sarrera eta irteera pareak osatuko ditugu ⁀ikurrarekin banatuz. Amaitzeko, `eu.tsv` fitxategian gordeko dugu dialogoa.

Testuingurua kontuan hartzeko, dialogoko aurreko interakzioa ere kontuan hartuko dugu. Beraz, sarreran 3 esaldi egongo dira, eta irteeran bat. Kasu honetan 100.000 lerro irakurri ditugu. Dialogo hau `eu_context.tsv` fitxategian gordeko dugu. Hone hemen testuinguruaren adibide bat:

1. Txanda:
   - Erabiltzailea: Kaixo.
   - Sistema:
     - Input -> Kaixo
     - Output -> Kaixo
2. Txanda:
   - Erabiltzailea: Zer moduz?
   - Sistema:
     - Input -> Kaixo Kaixo Zer moduz?
     - Output -> Ondo eta zu?
3. Txanda
   - Erabiltzailea: Ni ere
   - Sistema:
     - Input -> Zer moduz? Ondo eta zu? Ni ere
     - Output -> Pozten naiz.

Ingeleseko ereduan entrenamendurako datuak bakarrik erabili dira. Guk datuak hiru multzotan banatzea erabaki dugu, entrenamendua, balidazioa eta proba. Horrela, entrenatzen ari garen bitartean balidazioko datuetan probatu ahal izango dugu. Ondoren, probako datuak erabili ditzakegu emaitzak ikusteko. Datuen banaketa ausaz egin dugu, hurrengo tamainak erabiliz: entrenamendurako %80, balidaziorako %10 eta probarako %10.

5.2 Tokenizatzailea

Hiztegia sortu dugu Byte Pair Encoding (BPE) algoritmoa erabiliz (Sennrich et al., 2015). Defektuz 10000 subtoken definituko ditugu. Itzulpen automatiko neuronaleko ereduek hiztegi finkoarekin funtzionatzen dute normalean, baina itzulpena hiztegi irekiko arazoa da. Artikulu honetan, eredua hiztegi irekiko itzulpena egiteko gai da, hitz arraroak eta ezezagunak azpihitz unitateen kodifikatuz sekuentzia gisa.

Tokenizatzeko sarrera moduan aurreprozesatutako fitxategi osoa erabili dugu. Atzera begira, agian zentzu gehiago edukiko luke entrenamenturako datuak bakarrik erabiltzeak. Bestela, balidazioko eta probako datuak erabiltzen ari gara entrenamenduan eta horrek ereduari abantaila ematen dio. Horretarako, datuak tokenizatu aurretik banatu beharko lirateke, eta ondorekin hiru multzoak tokenizatu.

5.3 Eredua

Muturretik muturrerako sistema hau (Bahdanau et al., 2014) artikuluan proposatutako sisteman oinarritzen da. Hau itzulpen automatikoko sistema bat da, beraz, dialogoa itzulpen automatikoko ataza bat bezala definitzen ari gara. Aukera hau ez da optimoa sinplifikazio handi bat baita, hala ere, esperimentu interesgarriak egiteko aukera ematen digu.

5.3.1 Seq2Seq

Sekuentziatik sekuentziarako eredu (seq2seq) ohikoenak kodetzaile-deskodetzaile ereduak dira. Normalean sare errekurrente neuronal bat (RNN) erabiltzen dute sarrerako esaldia testuinguru bektorean kodetzeko. Bektore hau sarrerako esaldi osoaren errepresentazio abstraktua dela esan dezakegu. Bektore hau bigarren RNN batek deskodetzen du eta horrek irteerako esaldia ikasten du hitzak banaka sortuz.

Adibidez, ikusi 1 irudia. Sarrera esaldia, "guten morgen", embedding geruzatik (horia) igarotzen da eta ondoren kodetzailean sartzen da (berdea). Halaber, sekuentzia hasiera `<sos>` eta sekuentziaren amaiera `<eos>` tokenak eransten ditugu. Adibide honetan $X = \{x_1, x_2, ..., x_T\}$ daukagu, non $x_1 =$ <sos>, $x_2 =$ guten, etab. Hasierako egoera ezkutua, h_0, normalean zerora hasieratzen da edo ikasitako parametro batera.

Urrats bakoitzean, RNN kodetzailearen sarrera uneko hitzaren embeddinga txartatzen da, $e(x_t)$, baita aurreko denbora-urratsaren ezkutuko egoera ere, h_{t-1}, eta RNN kodetzaileak ezkutuko egoera

berria ematen du h_t. Ezkutuko egoera orain arteko esaldiaren irudikapen bektorial gisa har dezakegu. Kodetzailea horrela erreperesenta daiteke:

$$h_t = \text{EncoderRNN}(e(x_t), h_{t-1})$$

Orain gure testuinguru bektorea dugu, z, deskodetzen has gaitezke irteerako esaldia lortzeko, "good morning". Berriro ere, hasierako eta bukaerako tokenak gehitzen dizkiogu esaldiari. Urrats bakoitzean, RNN deskodetzailearen sarrera (urdina) uneko hitzaren embeddinga, $d(y_t)$ da, baita aurreko urratsaren egoera ezkutua ere, s_{t-1}. Hasierako deskodetzailearen ezkutuko egoera, s_0, testuinguru bektorea da, $s_0 = z = h_T$. Horrela, kodetzailearen antzera, deskodetzailea honela irudika dezakegu:

$$s_t = \text{DecoderRNN}(d(y_t), s_{t-1})$$

Deskodetzailearen hitzak bata bestearen atzetik sortzen dira beti. Beti `<sos>` erabiltzen dugu deskodetzailearen lehen sarrerarako, y_1. Ondorengo sarreretarako, $y_{t>1}$, batzuetan sekuentziako eagiazko hurrengo hitza erabiliko dugu, y_t eta batzuetan gure deskodetzaileak iragarritako hitza, $\hat{y}_{t-1}$. Honi teacher forcing deitzen zaio.

Gure eredua entrenatzerakoan edo probatzerakoan, badakigu zenbat hitz dauden helburuko esaldian, eta hitzak sortzeari uzten diogu. Inferentzian ohikoa da hitzak sortzen jarraitzea ereduak `<eos>` token bat itzuli arte edo hitz kopuru maximo batera iritsitakoan.

Aurreikusitako esaldia daukagunean, $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_T\}$, helburuko esaldiarekin konparatzen dugu, $Y = \{y_1, y_2, ..., y_T\}$, , galera kalkulatzeko. Ondoren, galera hori gure ereduko parametroak eguneratzeko erabiltzen dugu.

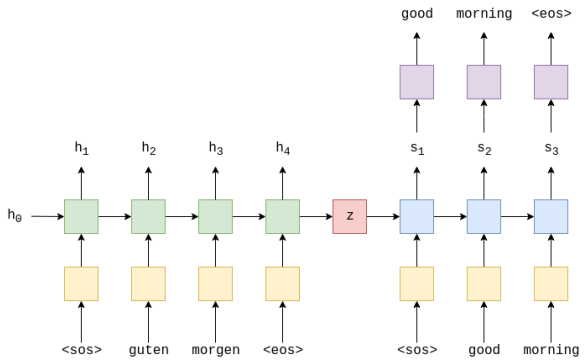

Figure 1: Eredua

5.3.2 Kodetzailea

RNN bidirekzionala erabiltzen dugu, geruza bakarreko GRU bat. Geruza bakoitzean bi RNN ditugu. Aurreranzko RNNtik esaldia ezkerretik eskuinera pasatuko da (berdez), eta atzeratutako RNNtik eskuinetik ezkerrera (urdinez). Ikusi 2 irudia. Hau horrela adieraz dezakegu:

$h_t^{\rightarrow} = \text{EncoderGRU}^{\rightarrow}(e(x_t^{\rightarrow}), h_{t-1}^{\rightarrow})$
$h_t^{\leftarrow} = \text{EncoderGRU}^{\leftarrow}(e(x_t^{\leftarrow}), h_{t-1}^{\leftarrow})$
non $x_0^{\rightarrow} =$ <sos>, $x_1^{\rightarrow} =$ guten eta $x_0^{\leftarrow} =$ <eos>, $x_1^{\leftarrow} =$ morgen.

Hasierako aurreranzko eta atzeranzko ezkutuko egoerak izango ditugu ($h_0^{\rightarrow}$ eta $h_0^{\leftarrow}$, hurrenez hurren). Bi testuinguru bektore ere lortuko ditugu, bata RNN aurreratua esaldian azken hitza ikusi ondoren, $z^{\rightarrow} = h_T^{\rightarrow}$, eta bestea RNN atzerakoia lehenengo hitza ikusi ondoren esaldian, $z^{\leftarrow} = h_T^{\leftarrow}$.

Aurreranzko eta atzeranzko ezkutuko egoerak konkatenatu ditzakegu, hau da, $h_1 = [h_1^{\rightarrow}; h_T^{\leftarrow}]$, $h_2 = [h_2^{\rightarrow}; h_{T-1}^{\leftarrow}]$. Egoera ezkutu guztiak horrela adieraz ditzakegu: $H = \{h_1, h_2, ..., h_T\}$.

Deskodetzailea ez denez bidirekziola bektore bakarra behar du, z, hasierako ezkutuko egoera gisa erabiltzeko, s_0. deskodetzailean bi ditugu, $z^{\rightarrow} = h_T^{\rightarrow}$ eta $z^{\leftarrow} = h_T^{\leftarrow}$. Hau konpontzeko, bi testuinguru bektoreak elkarrekin kateatu, g geruza lineal batetik pasatu eta $\tanh$ aktibazio funtzioa aplikatu dezakegu.

$$z = \tanh(g(h_T^{\rightarrow}, h_T^{\leftarrow})) = \tanh(g(z^{\rightarrow}, z^{\leftarrow})) = s_0$$

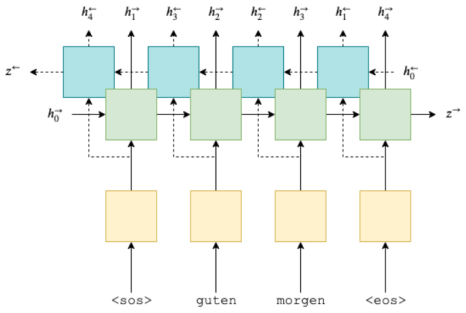

Figure 2: Kodetzailea

5.3.3 Atentzioa

Atentzio geruzak deskodetzailearen aurreko ezkutuko egoera hartuko du, s_{t-1}, eta kodetzailearen ezkutuko aurreranzko eta atzeranzko egoera guztiak, H. Geruzak atentzio bektorea itzuliko du, a_t, iturburu esaldiaren luzera duen bektorea. Elementu bakoitza 0 eta 1 artekoa izango da eta bektore osoaren batura 1. Bektore honek adierazten

du sarrerako esaldiko zer hitzi jarri behar diogun arreta hurrengo hitza aurreikusteko, $\hat{y}_{t+1}$.

Lehenik eta behin, aurreko deskodetzaile ezkutuko egoeraren eta kodetzaile ezkutuko egoeren arteko energia kalkulatuko dugu. Honek kodetzailearen ezkutuko egoera bakoitza aurreko deskodetzaile ezkutuko egoerarekin zenbateraino bat datorren adierazten du. E_t energia kalkulatuko dugu $attn$ geruza linealetik eta $\tanh$ aktibazio funtzio batetik pasatuz.

$$E_t = \tanh(\text{attn}(s_{t-1}, H))$$

Honek sarrerako esaldiaren luzera izatea nahi dugu. Horretarako, energia v bektorearekin biderkatuko dugu. Pentsa dezakegu v kodetzaileren ezkutuko egoera guztien energiaren baturaren pisuak direla. Pisu hauek adierazten digute zenbateko arreta hartu beharko genukeen token bakoitzari iturburu sekuentzian. v parametroak ausaz hasieratzen dira, baina gainerako ereduarekin atzera hedapenaren bidez ikasten dira.

$$\hat{a}_t = vE_t$$

Azkenik, atentzio bektoreak murriztapenak betetzen dituela ziurtatuko dugu, softmax geruzatik igaroz.

$$a_t = \text{softmax}(\hat{a}_t)$$

Adibidez, ikusi 3 irudia. Hau lehen arreta bektorea kalkulatzeko da, non $s_{t-1} = s_0 = z$. Bloke berdeek ezkutuko egoerak adierazten dituzte eta atentzioaren kalkulua bloke arrosaren barruan egiten da.

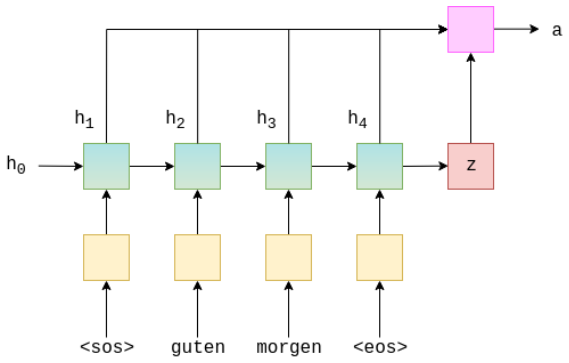

Figure 3: Attention

5.3.4 Deskodetzailea

Deskodetzaileak aurretik azaldutako atentzio geruza dauka barnean. Esan bezala, aurreko ezkutuko egoera hartzen du, s_{t-1}, kodetzaile ezkutuko egoera guztiak, H, eta arreta bektorea itzultzen du, a_t.

Atentzio bektore hau erabiltzen dugu iturri bektore pisatua sortzeko, w_t, , hau da, kodetutako ezkutuko egoeren batura pisatua, H, pisu gisa a_t erabiliz.

$$w_t = a_t H$$

Sarrerako hitzaren embeddinga, $d(y_t)$, iturburuko bektore pisatua, w_t, eta deskodetzailearen aurreko ezkutuko egoera, s_{t-1}, RNNra pasatzen dira, $d(y_t)$ eta w_t elkarrekin kateatuz.

$$s_t = \text{DecoderGRU}(d(y_t), w_t, s_{t-1})$$

Ondoren, $d(y_t)$, w_t eta s_t kateatu eta geruza lineal batetik pasatzen dira, f, irteerako esaldiko hurrengo hitza aurreikusteko, $\hat{y}_{t+1}$.

$$\hat{y}_{t+1} = f(d(y_t), w_t, s_t)$$

Ikusi 4 irudian adibideko lehen hiztaren deskodeketa. Bloke berdeek H itzultzen duten aurreranko eta atzeranzko RNNak adierazten dituzte. Bloke gorriak testuinguru bektorea adierazten du, $z = h_T = \tanh(g(h_T^{\rightarrow}, h_T^{\leftarrow})) = \tanh(g(z^{\rightarrow}, z^{\leftarrow})) = s_0$. Bloke moreak f geruza lineala adierazten du, $\hat{y}_{t+1}$ itzultzen duena. Bloke laranjak batura pisatuaren kalkulua egiten du, w_t, H eta a_t erabiliz.

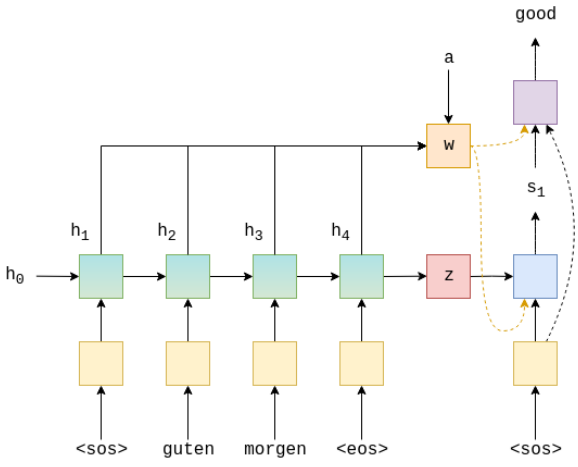

Figure 4: Decoder

5.4 Entrenamendua

Entrenamenduan hainbat aldaketa egin ditugu. Alde batetik, balidazioko pausoa gehitu dugu epoch bakoitzean. Bestetik, epoch bakoitzeko galera eta

perplexitatea gorde ditugu ondoren grafikak atera ahal izateko. Amaitzeko, ereduen checkpointak egin ditugu epoch bakoitzean, entrenamendua geratzen bada aurrerago jarraitu ahal izateko. Horretarako, ereduaren egoeraz gain optimizatzailearen egoera, epoch zenbakia eta aurretik esandako galera eta perplexitate balioak gorde ditugu. Izan ere, entrenatzeko denbora asko behar zen, eta oso zaila da dena segidan egitea.

Euskarako eredua 50 epoch entrenatu dugu 500.000 daturekin. Ikusi 5 eta 6 irudiak. Ikasketa kurba nahiko arraroa atera zaigu kasu honetan, salto arraroak daude checkpoint-ak kargatu ditugun lekuetan. Puntu horietan balidazioko galera jaitsi egiten da, eta entrenamendukoa igo. Hala ere, argi dago eredua overfitting egiten ari dela, entrenamenduko galera jaisten ari da eta balidaziokoa igotzen.

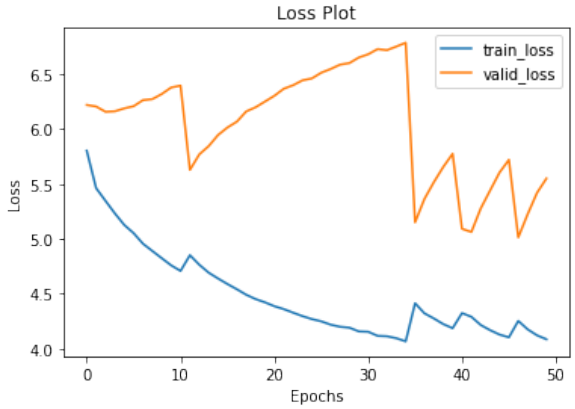

Figure 5: Loss

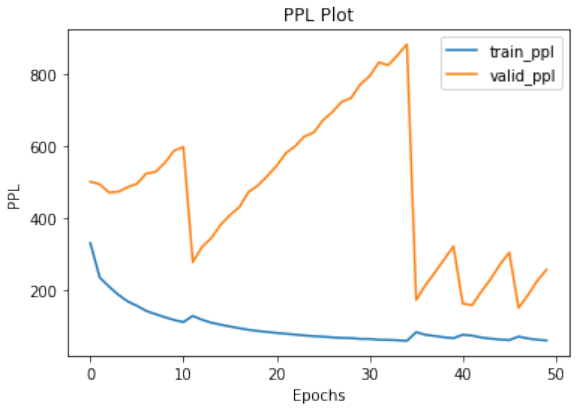

Figure 6: PPL

Testuingurudun eredua 50 epoch entrenatu dugu 100.000 daturekin. Ikusi 7 eta 8 irudiak. Kasu honetan ikasketa kurba normalagoa da. Izan ere, ez dugu checkpoint-ik erabili beharrik izan, entrenamendu denbora laburragoa zelako. Aurreko ereduak bezala overfitting egiten ari da. Datu gutxiago erabiltzen direnez, entrenamenduko galera azkarrago jaisten da, baina aldi berean balidaziokoa azkarrago igotzen da.

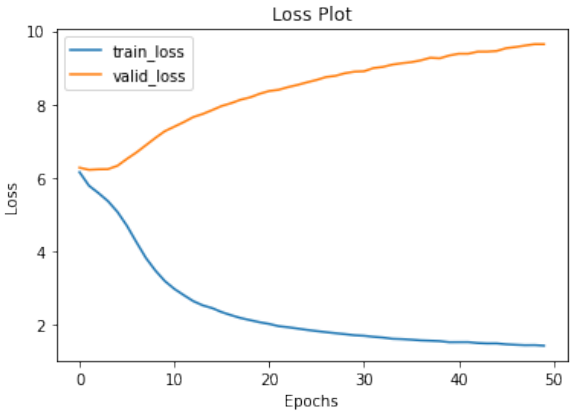

Figure 7: Loss Context

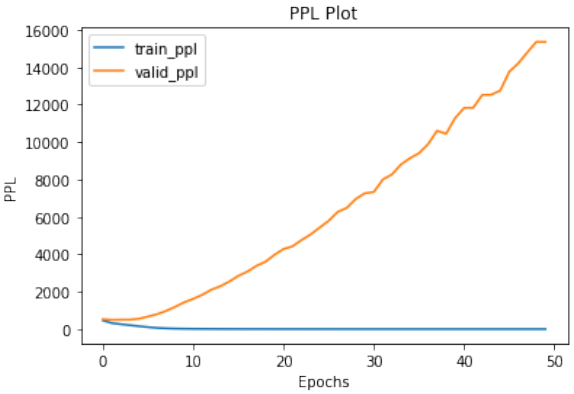

Figure 8: PPL Context

5.5 Inferentzia

Inferentziarako erabiltzailearen esaldia pasatzen zaio kodetzaileari. Guk esaldi hau garbitzea eta tokenizatzea erabaki dugu, entrenamenduko fitxategiarekin egiten den bezala. Testuingurudun ereduan sarrerako esaldiari testuingurua gehitu behar zaio. Horretarako, nahikoa da aurreko txandako balioa gordetzea eta hurrengo txandan ereduari pasatzea.

Ondoren, kodetzailearen emaitza eta hasierako tokena deskodetzaileari pasatzen zaizkio. Deskodetzailea erabiliz banaka hitz berriak sortu beharko ditugu. Baina deskodetzaileak itzultzen duena probabilitate distribuzio bat denez, hainbat estrategia daude hitza aukeratzeko.

Gure kasuan 3 greedy deskodeketa estrategia probatu ditugu: top1, topk eta multinomial. Top1

Figure 9: Hizkuntza aldatu

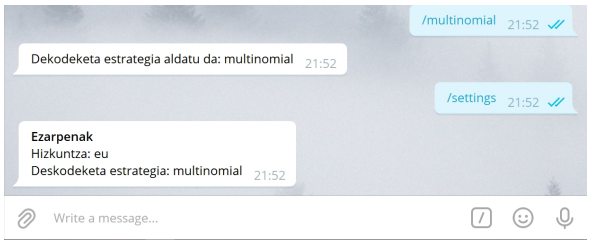

Figure 10: Deskodeketa aldatu

estrategiak beti token probableena aukeratuko du. Beraz, sarrera bererako beti irteera bera itzultzen du. Topk estrategiak k probableenen artean ausaz aukeratzen du. Multinomial estrategiak probabilitate distribuziotik lagintzen ditu tokenak. Tenperatura parametroa handitu daiteke probabilitate distribuzioa leuntzeko. Horrela, tenperatura baxua bada, zailagoa izango da probabilitate txikia duten tokenak aukeratzea.

5.6 Telegrameko txatbota

Txatbot bat, adimen artifiziala (AA) erabilita, erabiltzaile batekin lengoaia naturaleko elkarrizketa bat simulatu dezakeen aplikazio bat da. Gure kasuan, Telegrameko API-a erabiliz, txatbot bat sortu dugu. Txatbot honek, erabiltzaileak idatzitako mezua jaso eta aurrez entrenatu ditugun ereduak erabiliz, erabiltzaileari erantzuten dio. Txatbotak hainbat ezarpen desberdin ditu, hizkuntza zein aurreko azpisekzioan azaldutako deskodeketa estrategia bat aukeratu daiteke. Horretarako hainbat komando desberdin daude.

Alde batetik lengoaiarekin zerikusia duten komandoak daude:

1. **\eu** komandoak txatbotaren hizkuntza euskarara aldatzen du, hau da, euskarako eredua erabiliko du elkarrizketan.

2. **\en** komandoak txatbotaren hizkuntza ingelesera aldatzen du, hau da, ingeleseko eredua erabiliko du elkarrizketan.

3. **\context** komandoak ere txatbotaren hizkuntza euskarara aldatzen du, baina, testuingurudun euskarako eredua erabiliko du elkarrizketan.

4. **\setlanguage** komandoak botoi bidez hizkuntza aldatzeko aukera ematen du. 9 irudian ikus daiteke adibide bat.

5. **\language** komandoak txatbotaren uneko hizkuntza itzultzen du. 9 irudian ikus daiteke adibide bat.

Bestalde, deskodeketa estrategiarekin lotuta dauden komandoak daude:

1. **\top1** komandoak txatbotaren deskodeketa estrategia top1-era aldatzen du.

2. **\topk** komandoak txatbotaren deskodeketa estrategia topk-ra aldatzen du.

3. **\multinomial** komandoak txatbotaren deskodeketa estrategia multinomialera aldatzen du. 10 irudian ikus daiteke adibide bat.

4. **\setdecoding** komandoak botoi bidez deskodeketa estrategia aldatzeko aukera ematen du.

5. **\decoding** komandoak txatbotaren uneko deskodeketa estrategia itzultzen du.

Amaitzeko, beste bi komando ere badaude:

1. **\settings** komandoak uneko hizkuntza eta deskodeketa estrategia itzultzen ditu.10 irudian ikus daiteke adibide bat.

2. **\help** komandoak komando guztien lista itzultzen du.

6 Emaitzak

6.1 Inferentzia

Ingeleseko ereduarekin hainbat proba egin ditugu deskodeketa estrategia desberdinekin eta k eta tenperatura desberdinekin. Orokorrean ikusi dugu k eta tenperatura handitzean emaitza arraroagoak lortzen direla. Beste bi ereduetarako ere 3 aukerak probatu ditugu, baina k eta tenperatura defektuzko balioak erabiliz. Egindako probekin esango genuke eredu onena ingelesekoa dela, ondoren testuingurudun eredua eta azkenik testuingururik gabea.

Bestalde, deskodeketa estrategi desberdinekin lortutako emaitzen adibide batzuk jarraian ikus daitezke.

Deskodeketa multinomiala:

```
- Hello, how are you?
+ i ’ good and you  ?

- Hello, how are you?
+i ’m great .!
 i am a tour student.
```

Topk deskodeketa:

```
- Hello, how are you?
+ great how are you  like

- Hello, how are you?
+ ok just so what kind
  hows new shoes ? i
```

Top1 deskodeketa:

```
- Hello, how are you?
+ i ’m good great  .
  i ’m great  .
  what are you  ?
```

Gure ustetan hoberena multinomiala da, esaldiek ia beti erabiltzailearen sarrerarekin zerikusia izateaz gain, gramatikoki zuzenak baitira. Top1 eta topk-ren artean, top1-en emaitzak topk-renak baino konsistenteagoak dira. Topk-ak sortutako erantzun askok ez dute erabiltzaileak idatzitakoarekin zerikusirik, nahiz eta beste batzutan emaitzak onargarriak izan.

6.2 Atentzioa

Sortutako helburu token bakoitzaren ereduaren atentzioa erakusten duen funtzioa inplementatu dugu. Ebaluatzeko funtzioa erabiliko dugu iragarritako esaldia eta atentzioa lortzeko. Grafikoki erakusten dugu iturburuko esaldia x ardatzean eta iragarritako esaldia y ardatzean. Zenbat eta karratu argiagoa, orduan eta atentzio handiagoa eman dio ereduak iturburu-hitz horri helburu-hitz hori itzultzerakoan.

Sistema bakoitzerako adibide bana atera dugu, atentzioa nolakoa den ideia bat egiteko. Ikusi irudiak 11, 12 eta 13. Ingeleseko sistemaren eta beste bien artean aldea nabarmena da, dena zuria edo beltza baita. Euskarazko atentzioan gris desberdin asko daude. Atentzioetan fijatzen bagara, zaila da jakitea zenbaterainoko erabilgarria den. Itzulpenarekin konparatuta, dialogoan gutxiago laguntzen duela uste dugu. Izan ere, itzulpeneko atazan oso argi ikus daiteke atentzioa egokia al den. Baina, dialogoan sarrerarako hitzen eta irteerako hitzen lortura ez da horren zuzena.

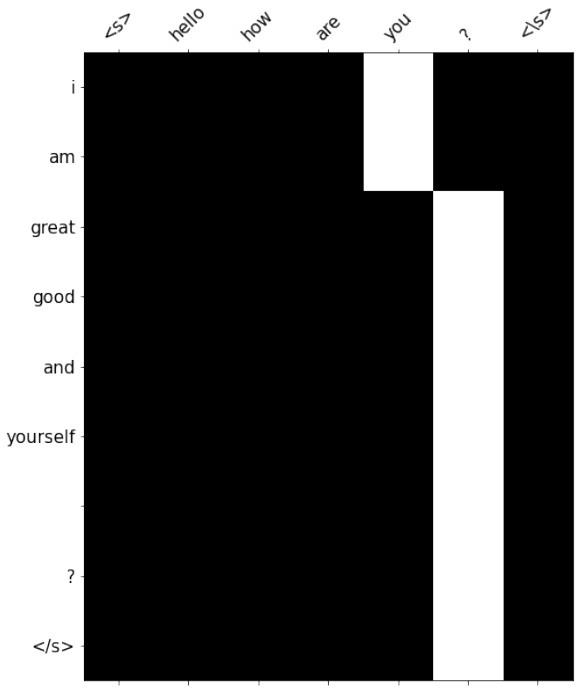

Figure 11: Atentzioa EN

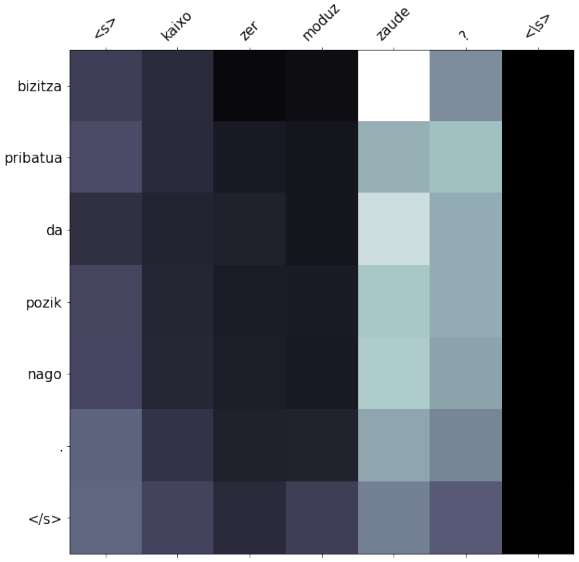

Figure 12: Atentzioa EU

6.3 Metrikak

Entrenamendua amaitutakoan, ereduen hainbat metrika kalkulatu ditugu eskuzko ebaluazioaren osagarri moduan. Dialogo sistemetan ebaluazio au-

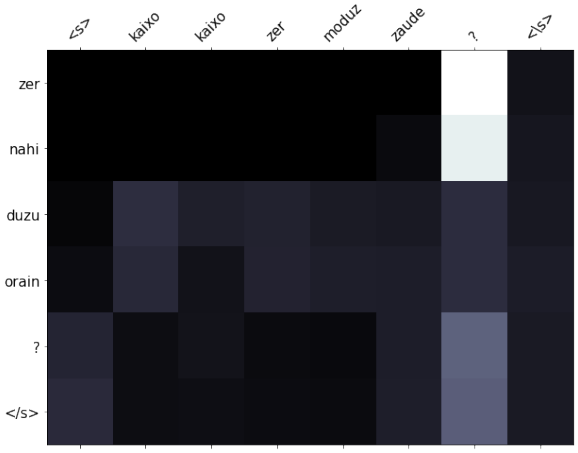

Figure 13: Atentzioa EU Context

	BLEU
Train EN Top1	29.78
Train EN Topk	5.17
Train EN Multinomial	29.85

Table 1: Decoding Strategy BLEU

	Loss	PPL	BLEU
Train EN	1.686	5.400	29.85
Train EU	4.678	107.558	4.00
Validation EU	4.675	107.263	1.53
Test EU	4.689	108.763	2.20
Train Context	2.892	18.024	24.29
Validation Context	2.880	17.809	25.21
Test Context	2.882	17.853	24.18

Table 2: Metrikak

tomatikoa ez da oso esanguratsua eta errealitatean beti egiten da eskuzko ebaluazio bat gizakiekin. Hala ere, metrikak erabiltzea ez dago soberan.

Entrenatzerakoan ereduaren galera edo perplexitatea baino ez zitzaigun axola. Hala ere, itzulpenaren edo dialogoaren kalitatea neurtzeko bereziki diseinatutako metrikak daude - ezagunena BLEU da. BLEUk aurreikusitako eta benetako sekuentzien gainjartzea aztertzen du bere n-grametan. 0 eta 1 arteko zenbaki bat emango digu sekuentzia bakoitzeko, non 1-ek gainjartze perfektua dagoen esan nahi duen. Hala ere, normalean 0 eta 100 artean agertzen da. BLEU iturburu sekuentzia bakoitzeko hautagaien itzulpen anitzetarako diseinatu zen, baina datu multzo honetan iturri bakoitzeko hautagai bakarra dugu.

Hasteko, ingeleseko entrenamenduko fitxategirako BLEU kalkulatu dugu deskodeketa estrategia desberdinak erabilita. Ikusi 1 taula. Bertan ikus daiteke top1 eta multinomial estrategiekin emaitza askoz hobeak lortzen direla. Eta kontuan hartuta top1 estrategiak beti emaitza bera itzultzen duela, gure ustez multinomial estrategia da onena.

Argi geratu da deskodeketa estrategiak ere eragin handia duela azken sistemaren kalitatean. Izan ere, errealitatean baseline bezala beam search algoritmoa erabiltzen da decoding-a egiteko eta guk erabiltzen ditugun greedy estrategiak oso sinpleak dira, decoding pauso bakoitzean bide bakarra mantentzen baitute.

Ondoren, multinomial estrategiarekin egin ditugu gainerako probak. Ikusi 2 taula. Sistema guztietako datuen azpimultzo bakoitzerako galera, perplexitatea eta BLEU kalkulatu ditugu. Emaitza onenak ingeleseko sistemak lortu ditu eta ondoren testuingurudun sistemak. Hori bai, esan beharra dago agian datu hauek ez direla oso esanguratsuak. Izan ere, 3 sistemak eredu eta datu desberdinekin entrenatu dira. Hala ere, idia bat egiteko behintzat balio digu.

Euskerako bi ereduen arteko aldea oso handia da. Honen arrazoia gure ustez erabilitako datu kopurua da. Testuinguruko ereduan 5 aldiz datu gutxiago erabli ditugu. Agian datu gehiegi zeuden ereduaren konplexutasuna kontuan hartuta eta horregatik kostatu zaio gehiago emaitzak hobetzea testuingururik gabeko ereduari. Ez dugu uste testuinguru sinple hau gehitzeak ereduari abantaila handirik ematen dionik.

Emaitzetan harritu gaitu eredu bererako datumultzoan artean ia alderik ez egoteak. Honen arrazoietako bat izan daiteke entrenamendu garaian egiten den teacher forcing-a, baina horrekin bakarrik ezin daiteke azaldu. Datu-multzo desberdinekin probak egitearen helburua zen ikustea zenbateko aldea dagoen entrenamenduko eta balidazioko puntuazioen artean. Entrenatzen ari ginen bitartean balidazioko emaitzak entremendukoak bana askoz txarragoak ziren. Epoch gehiago egin ahala entrenamenduko galera jaitsi egiten zen, eta balidaziokoa igo, hau da, eredua overfitting egiten ari zen.

7 Ondorioak

Lan honetan dialogoa itzulpen automatikoko ataza bat bezala defini dugu. Esan bezala, aukera hau ez da optimoa sinplifikazio handi bat baita. Beraz, hasieratik mugatuta geunden eredu mota dela eta. Horrez gain, argi geratu da datuek duten garrantzia.

Ingeleseko sistema entrenatzeko, pelikulen azpitituluak erabili beharrean elkarrizketak erabili ziren. Azkenengo hauek askoz aproposagoak dira ataza honetarako eta emaitzek garbi islatzen dute hori.

Erreferentziak

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.