BertsoBOT: lehen urratsak

Egilea: Manex Agirrezabal Zabaleta
Tutoreak: Iñaki Alegria, Bertol Arrieta
Kolaboratzailea: Aitzol Astigarraga

HIZKUNTZAREN AZTERKETA ETA PROZESAMENDUA

Hizkuntzaren Azterketa eta Prozesamendua Masterreko titulua lortzeko bukaerako proiektua

2012ko iraila

Sailak: Lengoaia eta Sistema Informatikoak, Konputagailuen Arkitektura eta Teknologia, Konputazio Zientziak eta Adimen Artifiziala, Euskal Filologia, Elektronika eta Telekomunikazioak.

Laburpena

Hizkuntzaren prozesamenduko teknikak erabilita, poesia-sorkuntza automatikoan lehen urratsak eman dira. Hau erdiesteko, corpusen prozesamenduan oinarritutako bilaketak erabili dira, bai bilaketa arruntak eta baita bilaketa semantiko aurreratuak ere, horretarako IXA taldean garatutako tresna ezberdinak erabiliaz. Hizkuntza poetikoko testuek, gramatikaltasun eta metrika hertsitik haratago, semantika eta pragmatika barneratuta dituzte. Lan honetan semantikaren auziari heldu zaio nagusiki.

Abstract

In this text, I present the first steps in computatinal linguistics, to allow the automatic generation of poetry. In order to achieve it, different corpora search techniques have been used, from simple string-match searches, to advanced semantic searches, using different tools developed by the IXA group. Poetic language is more than simple metrics and gramaticality, as it has plenty of semantical and pragmatical information. In this work we focused on semantics.

Gaien aurkibidea

Irudien zerrenda

1 Proiektuaren nondik norakoak

Hizkuntzaren Azterketa eta Prozesamendua masterraren baitan, corpusen prozesamenduan eta informazioaren berreskurapenean oinarrituta, bertsoak automatikoki sortzeko lana egin dugu. Helburu hau lortzeko bidean, hainbat teknika erabili ditugu: informazio-berreskurapena, corpusen prozesamendua, silabizazioa, ...

Proiektu honen aurrekari nagusiak BertsolarIxa web-aplikazioa (Arrieta et al., 2001) eta *Bertsotarako Arbel Digitala* (Agirrezabal et al., 2012a) dira. Lehenengoan errima-bilatzaile bat garatu zen euskararen morfologia nolabait alderantzikatuz. Arbel digitalaren proiektuan, berriz, bertsoaren egiturari dagozkion ezaugarriak tratatu genituen (errima- eta sinonimo-bilatzailea, errima-egiaztatzailea ...).

Lan honetan, bestalde, **bilatzaile semantiko** bat garatu dugu, hainbat atazatarako baliagarri izango zaiguna. Horretaz aparte, aurretik PHP lengoaiaz garatutako tresna batzuk, **egoera finituko teknologia erabilita berrinplementatu** ditugu (Agirrezabal et al., 2012b). Azken lan hau Donostian burututako FSMNLP[1] bilkuran aurkeztu genuen eta arbel digitalean txertatzear daukagu. Gainera, poesia- edo **bertso-sorkuntza automatikoaren inguruan lehen pausoak** eman ditugu, IXA taldean garatutako baliabide ezberdinak erabilita.

Lan hauek beste eremu batzuetako taldeen lanekin uztartu ditugu, hala nola, *Robotika eta Sistema Autonomoak*[2] eta *Aholab Signal Processing Laboratory*[3] taldeekin. Elkarlan honen ondorioz, 2012ko apirilean robot bat bertsotan jartzea[4] lortu zen.

Egin dugun lanean, sistema informatikoa hainbat ariketa ebazteko gai izan da corpusetan eta corpus horien gaineko bilaketetan oinarrituta:

- Bertso klasiko bat abestu
- Lau oinak emanda bertsoa abestu
- Hitz bat emanda bertsoa abestu
- Gai librean bertso bat abestu

Aipatutako aurrekarietan hitzen formari soilik heldu zitzaion, baina hauetatik haratago, bertsolari batek gramatikaltasuna, pragmatika eta munduari buruzko jakinduria izan ohi du. Etorkizunean bide hauek jorratu beharko dira, eta horretarako beharrezkoa izango da proiektu honetan sortutako bilatzaile semantikoa.

[1] Finite-State Methods and Natural Language Processing
[2] `http://www.sc.ehu.es/ccwrobot/`
[3] `http://aholab.ehu.es`
[4] `http://paperekoa.berria.info/plaza/2012-04-19/040/001/zientzia_fikzioa_bertsotan.htm`

Testu hau hainbat ataletan bereizi dugu. 2. atalean, irakurlea gai honetan murgiltzeko, artearen egoera zein den azaltzen da, orain arte eta gaur egun garatze-bidean dauden lanetan oinarrituta. Honen ondoren, eskura ditugun eta erabili ditugun baliabide batzuk aurkeztuko ditugu, modu laburrean. Gero, bertsolari-sistema osoaren arkitektura zein den plazaratuko da. Bilatzaile semantikoaren barrenak aztertu ondoren, goian aipatu ditugun ariketak nola burutzen dituen azaltzen da. Amaitzeko, gure sistemen ebaluazioa eta ondorioak erakutsi ondoren, etorkizunean jorratuko ditugun adarrak azalduko ditugu.

Bertsolaritza: jendaurreko ekintza soziala

Euskal Herrian bertsolaritza da ahozko literaturaren erakusgai behinena. Bertsolaritza ahozkotasunean eta bat-batekotasunean oinarritzen da, nahiz eta azken urteotan bertsolaritza idatziak nabarmen egin duen gora. Bertsolaritzaren inguruan asko idatzi bada ere (Gartzia et al., 2001), (Amuriza, 1981a) eta (Amuriza, 1981b) liburuak nabarmenduko nituzke. Aipatzekoa da baita 2011 urtean Asier Altunak zuzendutako "bertsolari" filma[5] [6].

Bertso bat zer den ulertzeko, Xabier Amurizak bertso batean emandako definizioa irakurtzea besterik ez dago:

Neurriz eta errimaz
kantatzea hitza
horra hor ze kirol mota
den bertsolaritza

Definizio horretan ikus daitekeen legez, bertsoa neurtuta eta errimatuta egiten den jardun kantatua da. Bertsotan, neurtzea esaten denean hiru elementuri egiten zaie erreferentzia: **lerro kopuruari**, lerro bakoitzeko **silaba kopuruari** eta lerro batzuek beste batzuekin **errimatu** egin behar izateari. Hona hemen adibide bat:

[5] `IMDB: http://www.imdb.com/title/tt2058583`
[6] `Trailer: http://vimeo.com/9355066`

Iparragirre abila dela (10)
askori diot aditzen (8)
eskola ona eta musika (10)
hori hoiekin zerbitzen (8)
ni ez nauzu ibiltzen (7)
kantuz dirua biltzen (7)
komediante moduan (8)
debalde festa preparatzen det (10)
gogua dedan orduan (8)

Xenpelarren bertso klasiko honetan argi ikusten da aurreko definizioaren funtsa. Aipatu beharra dago bertso honek darraien neurria zein den: bederatzikoa, handiaren moldekoa, sei puntuz. Honenbestez, bederatzi lerro izango ditu bertsoak eta handiaren moldekoa izatean, handien egituraren antzekoa izango du (lehen lau lerroek eta azken biek bat-egiten dute handien silaba-egiturarekin)[7]. Gainera, sei lerrotan agertuko zaigu errimatzeko beharra; kasu honetan, bi errimako multzoak daude (A eta B): 2, 4, 5 eta 6. lerroek elkarrekin errimatzen dute (A) eta 7. eta 9.ak ere bai (B). Errima-patroiak gorriz eta urdinez markatuta daude. Horretaz aparte, lerro bakoitzeko silaba-kopurua ere neurriak zehazten du (kolore berdearekin adierazia).

Neurri egitura hau, bertso-saio eta ekitaldietan ohikoa da. Hala izan zen, adibidez, 2011 urtean Gipuzkoan jokatu zen **Gpuntua** txapelketan, izan ere, ariketa batean neurri hori, eta espreski Xenpelarren bertso honen doinua, erabili behar izan zuten. Bertsotarako neurrien inguruan irakurri nahiz gero, gomendagarria da (elkarteko ikerkuntza taldea, 2009) liburua.

Neurri batek ezarritako mugak, lehen adierazi bezala, maila konputazionalean tratatu ditugu arbel digitalaren proiektuaren baitan. Bertan, silaba-kontatzaile bat ezarri genuen, horrela bertso bateko lerroak idatzi eta berehala, silaba kopuruak ea egokiak zirenentz adierazten zuen aplikazioak. Gainera, errima-bilatzaile bat jarri genion, euskararen morfologiaren deskribapenean (Alegria et al., 1996) oinarrituz, eta emaitzetan agertuko ziren hitzek gehienez bi atzizki izango zituztelarik. Azkenik, Euskal WordNet-en (Pociello, 2008) oinarrituta, sinonimo-bilatzaile bat ere txertatu genuen.

Baina bertsolaritza, ahozkotasunean oinarritzen den jendaurreko ekintza da. Beraz, ez da nahikoa izango bertsoak sortzea, sortutako poema horri *performance* bat gehitu beharko zaio. Horretarako, bertsoak sortzeko lanari beste lan batzuk gehitu behar izan zaizkio, hala nola, robot itxurako gorputz fisiko bat eta ahots sintetizatu bat.

[7]Neurri handietan lerro bakoitiek 10 silaba izaten dituzte eta bikoitiek 8

2 Artearen egoera

Poesia-sorkuntza automatikoaren inguruan hainbat lan egin dira urte hauetan. Horietariko askok ikasketa automatikoa (IA) baliatzen dute. Teknika estatistikoek hizkuntzaren prozesamenduan aurrerapauso handia suposatu dute, izan ere, automatizatzen lagundu dute duela gutxi arte eskuzkoa zen lana.

Bertsoak sortzerakoan bertsolaria ezin da metrikaz zuzenak diren esaldi soilak sortzera mugatu. Entzulea erakartzeko hainbat teknika erabili beharko ditu, hala nola, baliabide literarioak, txisteak ... Ikertzaile batzuk sormen konputazionalaren inguruan egindako lanek garrantzi handia izan dute.

Ondorengo ataletan, sorkuntza poetikoaren eta giza-sormenaren inguruan egin diren lan batzuk aipatzen dira:

2.1 Oulipo eta Alamo

Sorkuntza poetikoaren automatizazioaren garaia XX. mendean hasten da, informatikaren garapenarekin batera. Diziplina honen aitzindaritzat, idazle eta matematikari frantziarrak jotzen dira. Matematika, konbinatoria eta literatura bateratzeko bilkuren ondorio izan dira *Oulipo* eta *Alamo* bezalako taldeak.

Talde hauek, matematika eta literatura bateratzen dituzte poesia modu automatikoan sortzeko. Hainbat lan egin badituzte ere, nabarmenena edo ezagunena **Rimbaudelaires** da. Haien hitzetan, honen emaitzak ez dira ez *Rimbaud* eta ezta *Baudelaire*-en poesiak, bien arteko konbinazio bat baizik. Lehenik eta behin, *Rimbaud*-en poesia hartzen dute oinarritzat eta ondoren bertako izen, aditz eta adjektiboak *Baudelaire*-en testuetako hitzekin ordezkatzen dituzte.

2.2 Pablo Gervas

Mundu honetara jauzi egiten duen edonork, interneten bilaketa batzuk eginda, topo egingo du zientzialari honekin. Gervasek lan zerrenda handia dauka poesia sorkuntza konputazionalaren eremuan, eta hemen lan horietako bi aipatuko dira.

HAP masterra

2.2.1 WASP

Sistema honek (Gervás, 2000) sarrera gisa hitz zerrenda bat eta esaldi patroi batzuk emanda, poema multzo bat itzultzen du. Poema hauek. **Sortu eta test**[8] (*Generate & test*) metodoaz egiten du lan. Honen emaitzak metrikaren ikuspegitik perfektuak dira, baina ikuspegi linguistikotik, eskas samarrak eta zentzu gutxikoak. (Cañas eta Tardón, 2010) liburuan WASP erabilita sortutako poema baten adibidea agertzen da (gazteleraz):

Yunques ahumados
Sus muslos se me escapaban como
Peces sorprendidos
La mitad llenos de alas.
Con la sombra levanta
La arquitectura del humo
Un pie de mármol afirma
Su casto fulgor enjuto

2.2.2 ASPERA

Honek (Gervás, 2001) kasu bidezko arrazoitzearen[9] (*case-based reasoning*) metodoa erabiltzen du lan egiteko, eta NASAren *Clips* erregela-sisteman dago garatuta. Gervásen arabera, poesia-sorkuntzak bi pauso nagusi ditu: adierazi nahi den mezuaren sorkuntza eta mezu horren edertzea, baliabide ezberdinak erabilita. Bizitza errealean, pauso hauek batera egiten ditugu konturatu gabe, baina, Gervásen iritziz, soilik bigarrena da konputazionalki garatu daitekeena. Sistema hau gai da, mezu arrunt batetik abiatuta, mezu poetiko bat lortzeko. Honako datuak eman behar zaizkio sistemari poesia sortzeko: estilo poetikoan adierazi nahi dugun esaldia, poemak jarraitu beharreko egitura metrikoa, egitura metriko horrekin bat datozen kasu batzuk (hauek kasuen corpusean egon beharko dira) eta poeman erabili ahal izango liratekeen hitz solte batzuk. Bere lan-pausoak kasu bidezko arrazoitzearen urratsak dira:

- Lehenik eta behin, corpus batean mezu horrentzako ***case* berezitu bat berreskuratzen da** (*Case retrieval*), hau da, adierazi nahi den mezuaren antzeko esaldi bat bilatzen da.

[8] Sortu eta test: Problemak ebazteko metodo bat da. Metodo honen arabera, problemei soluzioak bilatzen zaizkie eta horiek probatu. Zuzenak ez diren kasuan, zuzenketa batzuk aplikatu eta berriro probatzen dira soluzioa topatu arte.

[9] Eranskinetan kasu bidezko arrazoitzearen teknikaren azalpen bat dago.

- Hau egin eta gero, aukeratu den *case* horren esaldi poetikoko POS[10] etiketen egitura jarraituta, hitz berriak jarrita, **esaldi berria sortzen da** (*CBR Reuse Step*), esaldi poetikoa, alegia.

- Sortzen den zirriborroa erabiltzaileari erakusten dio, hark **ebaluatu edo zuzentzeko** (*CBR Revise Step*).

- Amaitzeko, sortutako (eta kasu batzuetan zuzendutako) poema hau analizatu eta **corpusean txertatzen** du, gero lehen urratsean baliatzeko (*CBR Retain Step*).

Hau hobeto ulertzeko, Gervásen adibide batekin azalduko dugu. Suposa dezagun, poema bihurtu nahi dugun esaldia, gazteleraz, hau dela:

```
bebed los vasos de vino antes de que el camarero cierre
```

Esaldi honen hitzen kategoria morfosintaktikoa ateratzen da lehenik:

```
bebed/VLPM2P los/ARTDMP vasos/NCMP de/DET vino/NCMS antes/ADVT de/DET
```

```
que/CQUE el/ARTDMS camarero/NCMS cierre/VLPM2P
```

Gure esaldiaren antzekoak bilatzen dira kasuen corpusean, baina abstrakzio maila bat lortu ahal izateko, kategoria morfosintaktiko antzekoa[11] duten kasuak berreskuratzen dira. 1 irudian ageri den kasua berreskuratzen da; izan ere, bere POS etiketen zerrenda eta hasierako esaldiaren POS etiketen zerrenda oso antzekoak dira.

```
VLPM2P     ARTDMP NCMP DET        (NCMMS) ADVT DET CQUE ARTDMS NCMS VLPM2P

VLPM2P DET ARTDMP NCMP DET ARTDFS NCMFS   ADVT DET CQUE ARTDMS NCMS VLPM2P
```

Beraz, kasua berreskuratu da. Kasua berreskuratzean poemaren bukaerako egitura morfosintaktikoa definituta gelditzen da. Honen ondoren, POS etiketen lekuan beharrezko hitzak jarri beharko ditugu, horretarako hasieran aipatutako hitz solteen zerrenda erabiliko dugu. 2 irudian dago ikusgai hitz zerrenda hori.

Behin hitzak jarrita, berrikuspen atalerako ordua da. Sistemak itzuliko zukeen poema 3 irudian ikus daiteke.

[10] *Part-Of-Speech*, kategoria morfosintaktikoa

[11] Oraingoz soilik egitura-antzekotasuna neurtzen du. Intentzioa dute antzekotasun semantikoarekin lan egiteko.

```
(case (n acmamq) (beg nil) (end nil)
  (wordsProse
    disfrutad de los placeres de la juventud
    antes de que el tiempo pase)
  (POStagsProse
    VLPM2P DET ARTDMP NCMP DET ARTDFS NCMFS
    ADVT DET CQUE ARTDMS NCMS VLPM2P)
  (wordsPoetry
    *line*
    coged de vuestra alegre primavera
    *line*
    el dulce fruto antes que el tiempo airado
    *line*
    cubra de nieve la hermosa cumbre
    *line*)
  (POStagsPoetry
    *line*
    VLPM2P DET ADJPOSFS ADJGFS NCFS
    *line*
    ARTDMS ADJGFS NCMS ADVT CQUE ARTDMS NCMS ADJGMS
    *line*
    VLPS3S DET NCFS ARTDFS ADJGFS NCFS
    *line*)
)
```

1 irudia: ASPERAko errepresentazio paraleloaren adibidea

```
(vocabulary
  tomad      embriagadora  ba_quica
  to_nica    bebida        rojo
  ganimedes  mesonero      hostelero
  posadero   hombre        feo
  ciegue     ira           divina     fuente
)
```

2 irudia: ASPERArako poesiak osatzeko hitz-zerrenda

HAP masterra

```
(sample_output
tomad de vuestra ba_quica bebida
el rojo vino antes que el hombre feo
ciegue de ira la divina fuente )
```

3 irudia: ASPERAk sortutako esaldi poetikoa

2.3 Poevolve

Sistema honen (Levy, 2001) oinarria sare neuronalak dira. Bere oinarrian bilakaera-algoritmoak erabiltzen ditu, bere izenak argi adierazten duen bezala (*[Poe]try + [evolve]*). Honen egileak sormen konputazionalean aurrerapausoak eman nahi zituen, eta horretarako poesia-sorkuntza automatikoari ekin zion. *Poevolve* sistemak, hasiera batetan, *limerick*[12] motako poemak sortzen zituen. Sare neuronalak erabilita, harrigarria dirudien arren, hainbat poema sortu ondoren, benetako sormena duela esaten du bere lanean. Poemak sortu ahala, sistemak poema hobeak egiten ikasten du. *Limerick*-ez aparte, beste poesia egitura batzuekin ere lan egin du. 4 irudian, *poevolve* erabilita automatikoki sortutako *haiku*[13] bat ageri da, ingelesez:

klutz break spelled died be
goat Jake hot helped spelled truck lines
chair taste nan chair me

4 irudia: Poevolve-n sortutako *haiku* bat

2.4 Ruli Manurung

Manurung-ek poesia sortu nahi izan zuen automatikoki. Horretarako, artearen egoeran zeuden lanak aztertu eta poema batek beharrezko zituen hiru ezaugarri zerrendatu zituen: esanahia (*meaningfulness*), gramatikaltasuna (*grammaticality*) eta poetikotasuna (*poeticness*).

Bere esanetan, bestelako lanetan ezaugarri horietako azpimultzo bat soilik betetzen zen, baina ez hiru ezaugarriak. Beraz, hori ebazteko McGonagall sistema garatu zuen (Manurung, 2003). Bere oinarrian, Poevolve-n bezala, bilakaera-algoritmoak erabiltzen ditu, baina bi lan hauek modu independentean garatu direla esaten du.

[12]Limerick: Poema bat da limerick bat, umoretsua, burutazio argi bat edota zentzu gabeko zerbait izan daitekeena. 5 lerroz osatuta daude eta errima-egitura oso zorrotza daukate (AABBA).

[13]Haiku: Japonierazko poesia mota tradizionala da, 5-7-5 silabako lerroetan antolatutako 17 silabek osatua. Deskribapen objektiboa egiten dute haiku idazleek beren lanetan, baina irakurlearengan zirrarak eragitea izaten da deskribapen horren azken helburua.

2.5 Kim Binsted

Kim Binsted-ek umore konputazionalaren inguruan garatu zuen JAPE (Binsted, 1996), txisteen sorkuntzarako eta analisirako makina (*Joke Analysis and Production Engine*).

Tesi honetan, Binsted-ek lan handia egin zuen baliabide literarioekin, adibidez, aliterazio eta errimekin. Txisteak sortzeko, hitzen ahoskeran eta sinonimoetan oinarritzen ziren, horretarako WordNet datu-base lexikala erabilita.

2.6 Oliviero Stock eta Carlo Strapparava

Aipatzekoak dira bikote honek sormen konputazionalean eta emozioekin eginiko lanak. Haien lan ezagunenetako bat Hahacronym (Stock eta Strapparava, 2005) da. Honek akronimo bat jaso eta haren bertsio umoretsua itzultzen du, multzo semantiko kontrajarrietako hitzak erabiliz (WordNet bezalako baliabideetan oinarrituta).

Adibidez, akronimo batek erlijioaren inguruko hitzen bat badauka, hori sexu edo teknologiaren inguruko hitz batekin ordezkatzen du. Hori bai, hitz berriaren lehen letra berdina izan beharko da, akronimoak berdina izaten jarrai dezan. FBI siglek *Federal Bureau of Investigation* esan nahi dute. Hahacronym lanak, honako hau proposatzen du: *Fantastic Bureau of Intimidation*.

2.7 Graeme Ritchie eta Judith Masthoff

Graeme Ritchiek umore konputazionalaren inguruan lan asko garatu ditu. Bere lan aipagarrienetako bat, beste batzuen artean, Binsted eta Manurung-ekin batera sortutakoa, STANDUP asmakizun sortzaile automatikoa[14] da. Programa honen oinarria, Binsted-en tesian dago. 1997an Kim Binsted, Helen Pain eta Ritchie-k egindako azterketa batean, JAPE programarekin sortutako testuak umeentzako onargarriak liratekeen asmakizunak zirela ondorioztatu zuten. 2003tik 2007ra Manurung-ek ideia hauek hartu zituen oinarritzat eta JAPE berrinplementatu zuen, Java programazio lengoaia erabilita. Honela, hizkuntzaren jolastoki bat (Waller et al., 2009) sortu nahi izan zuen, komunikazio-zailtasunak zituzten umeentzako.

[14] `http://www.abdn.ac.uk/jokingcomputer/home.shtml`

3 Eskura ditugun baliabideak

Atal honetan, gure lanetarako baliagarriak izan diren baliabideak zerrendatu eta azalduko ditugu.

3.1 Hizkuntzalaritza konputazionalaren inguruko lanak

Bertsolaritzaren inguruan lan asko egin diren arren, atal honetan, hizkuntzalaritza konputazionalarekin erlazionatutako lan batzuk ere nabarmenduko ditugu. Gehienbat, euskara edo bertsolaritzarekin erlazionatutako lanak agertuko dira, baina kasu batzuetan, lan eleanitzak ere aipatuko ditugu.

3.1.1 Euskararen deskribapen morfologikoa eta honen *foma* bertsioa

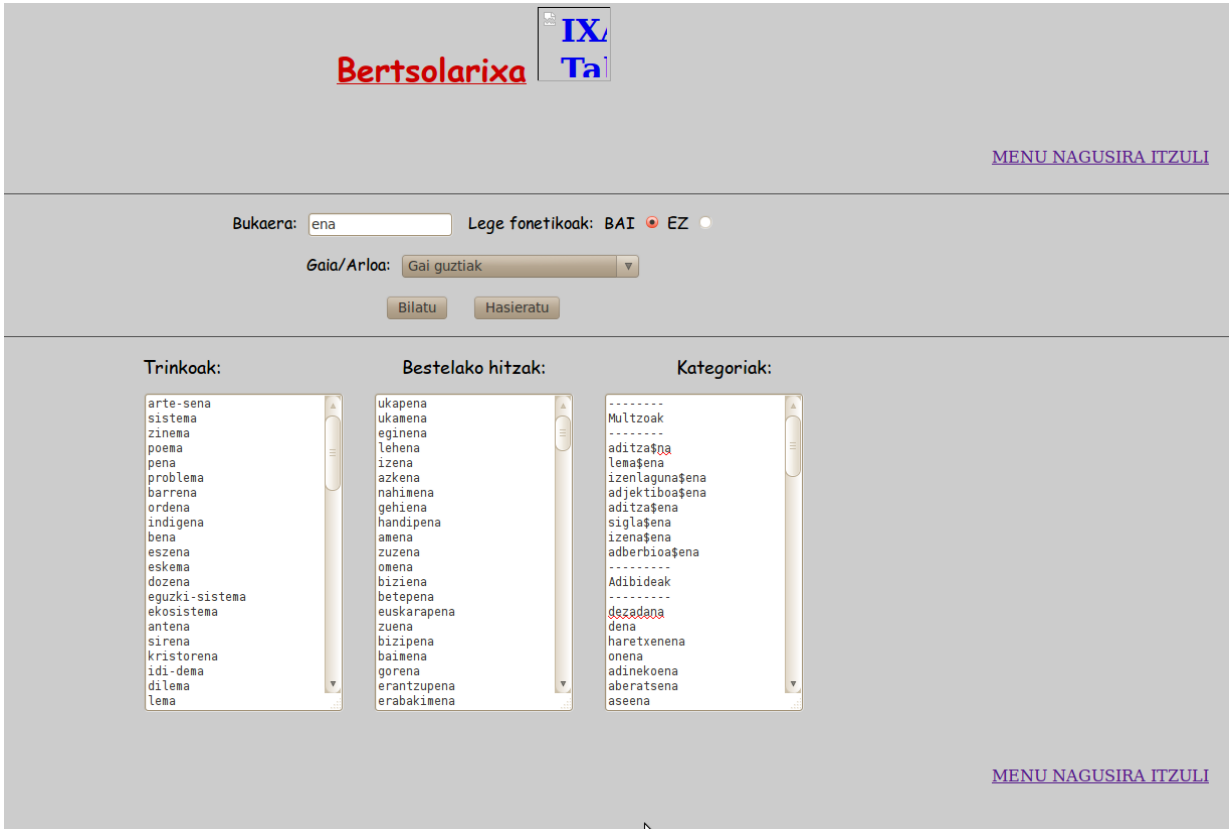
Lan honetan (Alegria et al., 1996), euskararen morfologia deskribatzen da egoera finituko teknologia erabilita, Xerox-en (Beesley eta Karttunen, 2003) tresnekin. Aurrerago *foma* (automatak eta transduktoreak konpilatzeko tresna ireki bat) garatu zuen Mans Huldenek (Hulden, 2009) eta hau erabilita, euskararen deskribapen morfologikoa *foma* formalismora pasatu zuten (Alegria et al., 2010).

Honi esker garatutako lanik garrantzitsuenetarikoa, Xuxen[15], euskararako zuzentzaile ortografikoa (Agirre et al., 1992) izango litzateke. Honetan oinarritzen da lan honetan erabili dugun errima-bilatzailea ere, "bertsotarako arbel digitala" proiektuaren baitan garatua.

3.1.2 BertsolarIxa

Euskarazko lehen errima-bilatzaile informatikoa[16] (Arrieta et al., 2001). Esan daiteke honekin eman zirela bertsolaritza eta hizkuntzalaritza-konputazionala biltzen zituen lehen urratsak. Bere oinarrian atzekoz aurrerako sorkuntza morfologikoa erabiltzen du, egoera finituko teknologietan oinarrituta. Atzekoz aurrerako sorkuntza morfologikoa egiteko, euskararen deskribapen morfologikoa alderantzikatu behar izan zen. Gainera, errimak bilatzerakoan, emaitzak multzokatu egiten ditu. Adibidez, "ena"-rekin errimatzen duten hitzak bilatuz gero, "pena" eta "eskema" bezalako hitzak itzultzen ditu. Baina gainera, gai da ["adjektibo" + "ena"] motako emaitzak itzultzeko. Honen lehen bertsioa Bertol Arrietak garatu zuen, eta gero, Oihan Odriozolak hobetu egin zuen.

[15] `http://www.xuxen.com`
[16] `ixa2.si.ehu.es/bertsolarixa/nagusia.html`



5 irudia: BertsolarIxa web-aplikazioaren interfazea

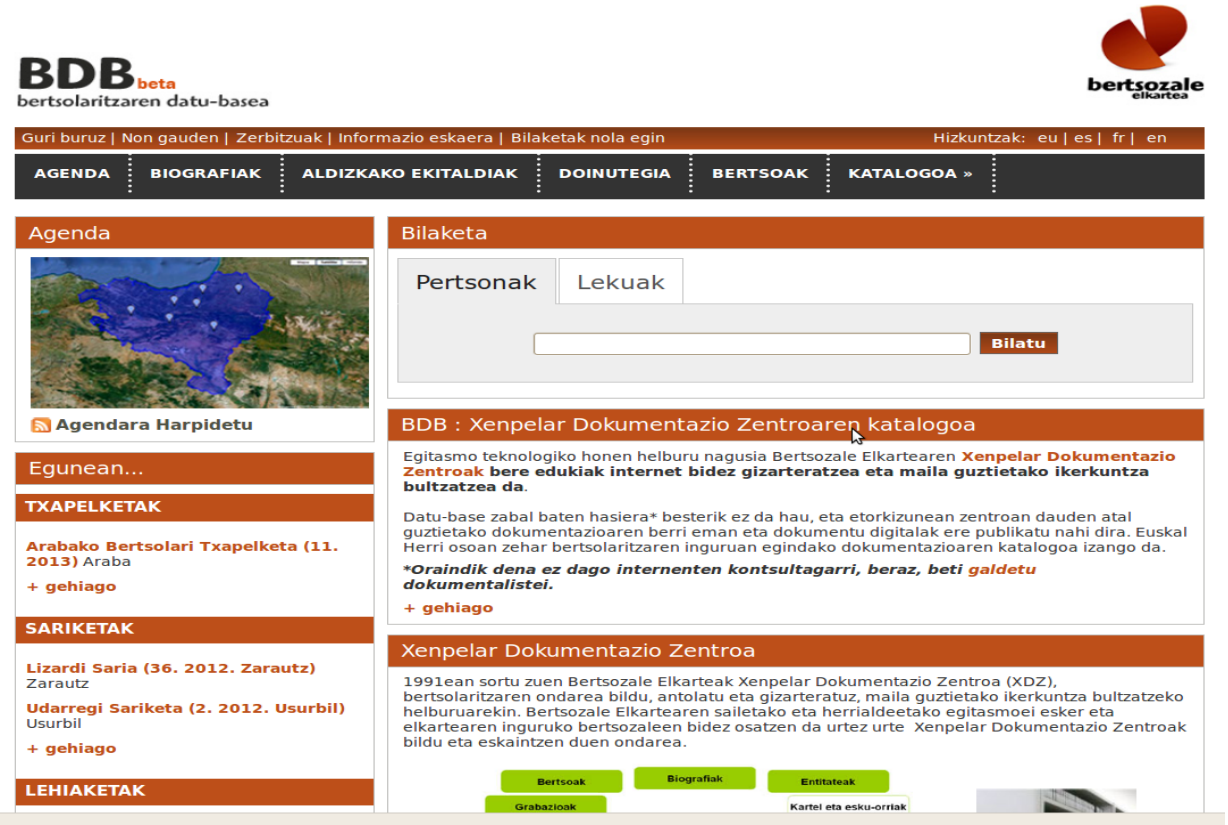
3.1.3 Bertsolaritzaren datu-basea

Xenpelar Dokumentazio Zentroak bertsolaritzarekin erlazionatutako orotariko materialaren bilketa eta katalogazioan egin du lan[17]. Ondorengo azpiatalean dagoen "Bertsotarako arbel digitala" proiekturako berebiziko garrantzia izan du, izan ere, haiek dute katalogatuta Joanito Dorronsorok bildutako bertso-doinutegiaren (Dorronsoro, 1995) bertsio digitala. Doinutegiaz gainera, bilduta dauzkate bertsolari txapelketa nagusietako bertsoen transkripzioak, eta bertsoen sorkuntzarako giltzarri izan dira hauek.

3.1.4 Bertsotarako arbel digitala

BertsolarIxa-ren inguruko hariei tiraka, Aitzol Astigarragaren ideia batean oinarrituta sortutako web-aplikazioa (Agirrezabal, 2011). Bertsotan ikasteko balio du eta bertsotarako ohiko neurri eta doinuak barneratuta dauzka, Xenpelar Dokumentazio Zentroaren eskutik.

[17] `http://bdb.bertsozale.com`



6 irudia: Bertsolaritzaren datu-basea

Tresna honek, gainera, silaba-kontatzaile bat, errima-bilatzaile bat, errima-egiaztatzaile bat eta sinonimo-bilatzaile bat ditu bere baitan.

Une honetan, arbel digitala bertso-eskolentzat eskuragarri jartzeko prozesuan gabiltza. Tresna baliagarria izango da hezkuntza inguruneetan, batez ere bertsolaritzaren transmisiorako. Gainera, etorkizunean hainbat modulu hobetzeko intentzioa dago: hala nola, arbelean idatzitako bertso bateko izenak, adjektiboak ... aztertuta, doinua automatikoki esleituko duen doinu-asignatzaile bat. 7 irudian web-interfazearen irudi bat dago ikusgarri.

Bertso-abeslaria

"Bertsotarako arbel digitala" proiektuaren baitan, bertsoa eta honen kantatzea uztartu ziren, honetarako *Aholab Signal Processing Laboratory*[18] ikertzaile taldearen inplikazioa berebizikoa izanik. Aholab taldeak, euskarazko ahotsaren tratamenduarekin egiten du lan, eta haien lanetako bat, euskarazko Text-To-Speech sistema baten garapena da (Hernaez

[18] `http://aholab.ehu.es`

7 irudia: Bertsotarako arbel digitalaren interfazea

et al., 2001). Bertsotarako doinuak erabilita, sistema honi (Erro et al., 2010) aldaketa batzuk egin eta makinak bertsoak abestea lortu da. Abesteko modulua web-zerbitzu gisa eskaintzea izan da proiektu honen barruan egin den ekarpen nagusia.

3.1.5 Informazioaren berreskurapena

Arantxa Otegik, 2012an, defendatutako doktorego-tesiaren (Otegi, 2012) ikerlerro nagusia informazio-berreskurapena izan da eta honi hedapen semantikoak egin diezaiokeen ekarpena. Bere emaitzak *Berbatek*[19] proiektuaren demoetan daude ikusgarri. Emaitza nagusia euskararako bilatzaile semantiko bat izan da, non behar den kasuetan hedapen semantikoa erabiltzen duen bilaketaren estaldura handitzeko (sinonimoak, hiperonimoak, hiponimoak ...). Oinarrian *mg4j* bilaketa-motorea erabiltzen du, bai bilaketak eta baita indexatzeak egiteko. Lan honetan oinarritu gara bertsoen bilatzaile semantikoa garatzeko.

3.1.6 Egoera finituko teknologia

Egoera finituko teknologia erabili da proiektu honetako lan batzuk garatzeko. Horretarako erabili dugun softwarea *foma*[20] (Hulden, 2009) da. *Foma* egoera finituko tresna multzo bat da, eta Lengoaia Naturalaren Prozesamenduan asko erabiltzen da, hala nola, analisi eta sorkuntza morfologikoan. Bereziki erabilgarria da morfologia aberatseko hizkuntzetan, adibidez, euskara.

[19] `http://www.berbatek.com`
[20] `http://foma.googlecode.com`

8 irudia: Bilatzaile semantikoaren demoaren interfaze nagusia

3.1.7 Egoera-finituko silabizazioa

Mans Huldenek 2006an egindako lanean (Hulden, 2006), egoera finituko teknologia erabilita, hitzen silabizazioa egiteko sistema bat garatu zuen[21]. Lan honen arabera, silaba bakoitza hiru zatiz osatua dago, *onset*, *nucleus* eta *coda*. Beraz, erregela ezberdinak erabilita, silaba bakoitzeko atalak hiru gelaxketako batean sartuta, hitzen silabizazioa egiten da. Lan honekin silaba-kontatzailea birprogramatu dugu. Honetarako, ingelesezko bertsioari aldaketa txiki batzuk egin behar izan dizkiogu euskararako baliagarria izateko. Orain artean, Aitzol Astigarragak *perl*-ez garatutako bat bageneukan, eta intentzioa dugu bien

[21] Silaba-banatzaile honen oinarrizko bertsioa `foma.googlecode.com` helbidean dago eskuragarri.

ebaluazio bat egiteko.

3.2 Robotikako lanak

Robotika eta Sistema Autonomoak taldearen[22] lanek ez dute hizkuntzalaritza konputazionalarekin zerikusi zuzenik, baina haiekin kolaboratu dugu proiektu honen barruan.

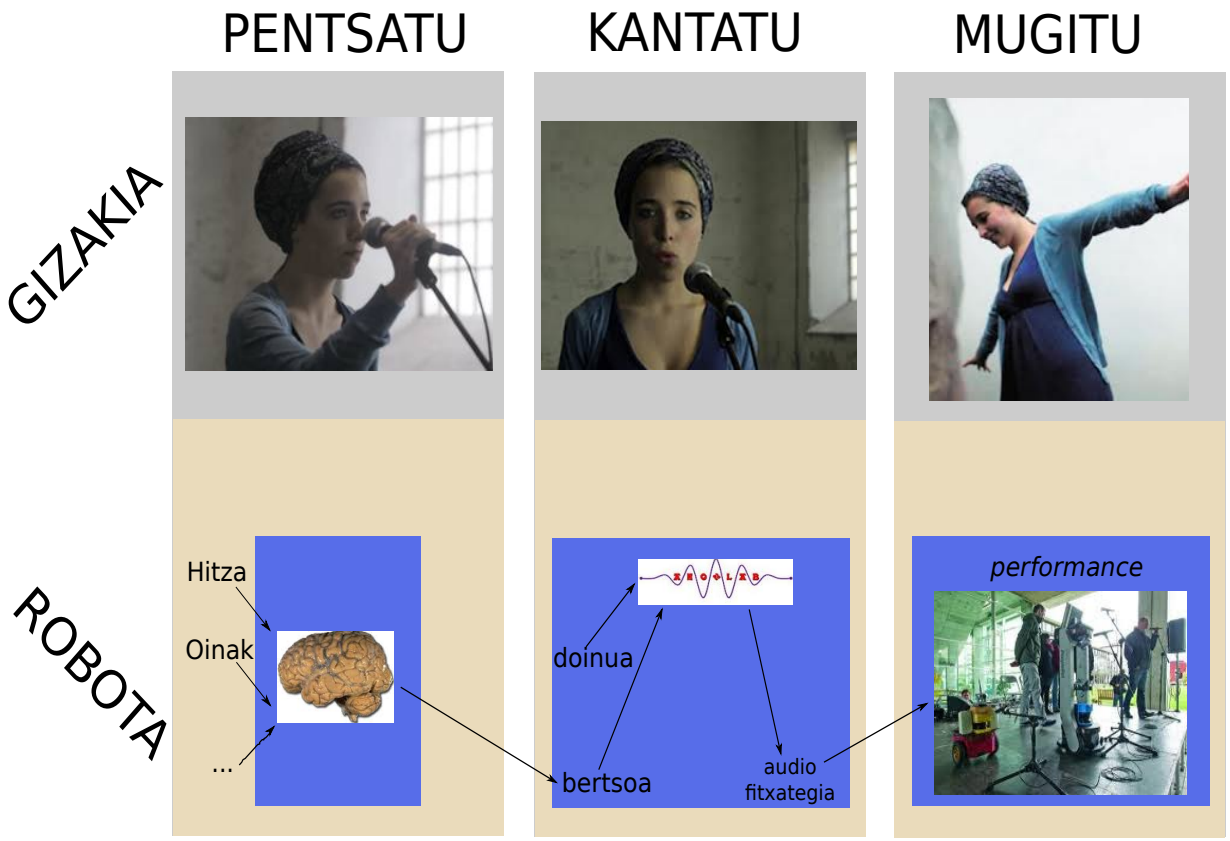
Talde honetan, robot mugikor eta autonomoekin egiten dute lan. Talde honen gogo eta lanari esker makinari gorputza jarri ahal izan diogu. Baina gorputz fisikoaz aparte, haien esperientzia mahai gainean jarri eta bertsolarien mugimenduak imitatu dituzte robotak erabilita.

Gure probetarako bi robot ezberdin erabili ditugu: Hurrenez hurren, *PeopleBot* eta *Pioneer* plataformadun Tartalo (Jauregi et al., 2007) eta Galtxagorri robotak (Lazkano, 2005).

[22] `http://sc.ehu.es/ccwrobot`

4 Sistemaren arkitektura eta diseinua

Atal honetan, robot-bertsolariaren hezurdura nagusia azalduko da, hau da, bere anatomia. Hau azaldu aurretik, sistemaren gorputzaren irudi bat ikustea lagungarri izan daitekeelakoan, 9 irudian, sistemaren arkitekturaren eskema ikus daiteke.



9 irudia: Gure sistemaren arkitektura

Argi uler daitekeenez, hiru modulu nagusi dauzka sistemak. Lehen modulua bertsoedo poesia-sorkuntza modulua da, non kantatuko duena "pentsatzen" duen. Hitz bat edo oinak jasota bertso bat itzultzen du. Hauxe da proiektu honetan nagusiki landu duguna. Bigarren moduluan, bertsoa kantatu egiten du, horretarako *Aholab*-eko ahoTTS modulu eraldatua erabilita. Azkenik, lan honen *performance*-a egiteko robotika lanak daude, bertsolari automatikoari giza-itxura eta giza-mugimenduak emateko.

Hiru modulu horien azalpen bat egiten da ondorengo hiru ataletan.

HAP masterra

4.1 Burmuina

Bi azpiataletan dago banatuta atal hau. Lehenik eta behin, sorkuntzarako oinarrizkoak diren corpusak nola antolatu ditugun azalduko da. Honen ondoren, bertsoak sortzeko bide ezberdinak erakutsiko ditugu.

4.1.1 Corpusen antolaketa

Lan honen garapenerako bi corpus ezberdin erabili ditugu. Lehenik eta behin, Xenpelar Dokumentazio Zentroak bildu eta sailkatutako bertso bilduma, Euskal Herrian egin diren bertsolari txapelketetako bertsoez osatua. Beste alde batetik, *Euskaldunon Egunkaria* euskarazko hedabidearen corpus bat ere eskura izan dugu eta erabili dugu[23]. Corpus bakoitzari erabilpen berezi bat eman zaio, eta horretarako egitura berezitu bat eman behar izan zaio.

Bertsoen corpusa, testu modura eskuratu eta haren XML bertsio bat sortu dugu, horretarako TEI5 etiketatze-araudi estandarra erabilita. Corpus honetan, bertsoen transkripzioekin batera, hainbat metadatu daude, hala nola, bertso-saioaren lekua eta data, saioaren informazioa, bertsoaldiaren egileak, burututako ariketa, etab..

10 irudian TEI5 araudia erabilita etiketatutako bertsoaldi bat dago. Bertsoak eskuz moztuta daude espazio gehiegi har ez dezaten.

*Euskaldunon Egunkaria*ren corpusari dagokionez, testu hutsezko corpusa oinarritzat hartuta, galbahe bat ezarri eta corpus berri bat sortu du Aitzolek. Silaba-kontatzaile automatiko batekin, corpuseko albisteetan ager zitezkeen silaba-egitura berezi batzuk erauzi ditu, 7 + 6 silabako egiturakoak hain zuzen (neurri txikien egitura). Egitura horiek erauzita eta bertsoen corpuseko zortziko txikiko estrofekin konbinatuta, 20.000 estrofa inguruko corpus bat lortu da.

Azpian zortziko txikiko corpusaren hamar lerroko lagin bat dago. Argi ikusten da 7 + 6 silabako egitura jarraitzen duela. Bertako azken hitza da bilatuko den oina.

```
Astea ez umore  ta ez pazientzi
zerbait merezi gendun  orren mutil finak
garizuman gaudenik  ez leike ahantzi
Buelta egin eta  bagatoz onuntza
Nere kolpea ez da  gaur izandu hutsa
Zerbait merezi zuan  neskatxa lirainak
probatu al dituzu  beraren ezpainak
Ederki gozatuaz  musu ta mingainak
nerbiuak pakean  ez didate utzi
Hola ibiltzen gera  batetik bestera
```

4.1.2 Bertso-sorkuntza lanak

Lan honetan bertso-sorkuntzari buruz aritzen garenean, bertsoen berreskurapenaz ari gara kasu gehienetan. Honekin esan nahi dena zera da, ez dugula hutsetik bertso zentzudunak

[23]Corpus honekin lortutako emaitzen meritu osoa Aitzol Astigarragarena da.

```
<Bertsoa trzenb='0'>
  <Kategoria>BAT-BATEKOA</Kategoria>
  <Saioa>2001-11-03 LESAKA TXAPELKETA</Saioa>
  <Gai-jartzailea></Gai-jartzailea>
  <edo-Gai-emailea>TABERNA-MIKEL</edo-Gai-emailea>
  <edo-Aurkezlea></edo-Aurkezlea>
  <Bertsolariak>IBARRA-IRATXE</Bertsolariak>
  <Gaia>Kristorenak eta bi pasatuta, patera zahar batean iritsi zinen ↙
     ↳ amesten zenuen lurraldera. Ailegatu eta segituan atxilotu ↙
     ↳ zintuzten. Orain, hegazkin dotore batean sartu eta berriz ere ↙
     ↳ zure lurraldera eramanen zaituzte.</Gaia>
  <Neurria>KLASIKOEN SAILA</Neurria>
  <Doinua>GAI HORREK BADU MAMIA</Doinua>
  <Lana>BAKARKA GAIA-EMANDA KARTZELAKO-LANA</Lana>
  <balorazioa-saioarena> Txapelketa Nagusia 2001. Final-laurdenetako ↙
     ↳ bost saiotatik laugarrena. Udal pilotalekuan, arratsaldeko ↙
     ↳ seiretan hasita, 450 bat entzule.
  </balorazioa-saioarena>
  <Bertsoaldiari-oharrak></Bertsoaldiari-oharrak>
  <Deskriptoreak> //  </Deskriptoreak>
  <Bertso-gaiburuak></Bertso-gaiburuak>
  <Gai-oharrak> </Gai-oharrak>
  <Argitaratua-Argitalpena></Argitaratua-Argitalpena>
  <Funtsa-Bilduma>Txapelketetako Bilduma</Funtsa-Bilduma>
  <Iturria>Xenpelar Dokumentazio Zentroa</Iturria>
  <Kokapena></Kokapena>
  <Transkribatzailea></Transkribatzailea>
  <Transkripzio-oharrak></Transkripzio-oharrak>
  <Izenbururako></Izenbururako>
  <Bertso-kopurua>2</Bertso-kopurua>
  <Transkripzioa>
    <lg zenb='1'> <!-- Line group: bertsoa-->
      <l>Panorama ona ez da</l> <!-- Line: lerroa-->
      <l>lurralde pobreena;</l>
      <l>bertan utzi nahi nituen</l>
      <l>hainbat kezka ta pena.</l>
      ...
    </lg>
    <lg zenb='2'>
      <l>Kaleetan barrena beti</l>
      <l>ume asko ortozik,</l>
      <l>eta gainera ez zen falta</l>
      <l>herrian ondoezik.</l>
      ...
    </lg>
  </Transkripzioa>
</Bertsoa>
```

10 irudia: TEI5 araudia erabilita etiketatutako bertso-transkripzio bat

sortzeko sistema bat garatu. Ariketa berezi batzuei bertso batzuekin erantzuteko gai den sistema bat egin dugu. Sortutako bertsoak, aurretik bertsolariek egindako bertsoak edo estrofa ezberdinak konbinatuta sortutako bertsoak izango dira orokorrean.

Aukera ezberdinak daude bertsoak berreskuratu edo bilatzeko. Batetik, bilaketa semantiko aurreratuak daude, non hitz bat bilatuta, bere lema eta senide semantikoen gainean ere bilaketak egiten diren. Bilaketa hauen inguruko datu gehiago 5.2 atalean ematen dira, aurrerago.

Baina bilaketa aurreratuetaz aparte, batzuetan interesgarriak izan daitezke string parekatze soilak. Adibidez, lau oinak emanda bertso bat itzultzea nahi denean. Oinen gainean bilaketak egiteko, 7 + 6 silabako estrofez osatutako corpusa erabili dugu, eta bertan, sei silabetako lerroan behar dugun oina bilatu (lerroko azken hitza).

Gerta daiteke oin batzuekin estrofarik ez topatzea. Hori gertatzen denean, bi estrategia ezberdin erabil daitezke; bertsorik ez kantatzea[24] edo edozein hitz edo soinurekin doinua kantatu eta bukaeran oina botatzea. Hona hemen bigarren irtenbidearen adibide bat:

la la la la la la la

la la kokakola

4.2 Kantuaren modulua

Jakina da bertsolaritza ahozko literaturaren adierazpen bat dela. Beraz, bertso-sorkuntza lanei kantatzeko gai den modulu bat gehitzea nahitaezkoa da. Horretarako, 3.1.4 atalean aipatutako bertso-abeslariaren modulua gehitu diogu, baina hori modu erdi-automatikoan erabili ahal izateko, hobekuntza nagusi bat egin behar izan diogu.

Orain artean, abeslariaren lana, laborategiko lan bat izan dela esan daiteke, izan ere, probatu ahal izateko zerbitzari batera konektatu eta script berezi batzuk exekutatu behar ziren. Une honetan, zerbitzari gisa erabil daiteke modulu hau web interfaze baten bidez[25], eta aukera dago gainera, *perl* bezalako programazio lengoaia baten bidez, bertso idatzi bat bidali eta *wav* fitxategi bat eskuratzeko. Hona hemen script horrek beharko zukeen oinarrizko zatia:

```
my $ua = LWP::UserAgent->new;
my $req = $ua->request(POST 'http://aholab.ehu.es/users/manex/abestu.php',
      Content_Type => 'form-data',
      Content => [
bertsoa => $bertsoa,
               melodia => $melodia[$zenb],
```

[24] Ostrukaren algoritmoa

[25] `http://aholab.ehu.es/users/manex`

```
                erritmoa => $erritmoa[$zenb],
                erritmoOsoa => '1',
                gizon => $ahotsa]);
my $url = 'http://aholab.ehu.es/users/manex/tmp/emaitza.wav';
my $fitx = 'bertsoa.wav';
getstore( $url, $fitx );
```

4.3 Robotikako lanak

Lanaren atal honek, dibertigarria dirudien arren, lan-ordu asko suposatu ditu. Robotikako arazo klasiko bat zera da; gauzak ez direla espero duzun bezala gertatzen gehienetan: izan ere, beti dago simulazio informatiko batean kontutan hartzen ez diren gauzak. Bertso-saio batean, adibidez, robotak aurrera eta atzera egin behar duenean, kable baten gainetik pasatzeak bere ibilbidea guztiz alda dezake.

Proiektu honen garapenerako, ez nuke eskertu gabe utzi nahi Elena Lazkano, Ekaitz Jauregi eta Aitzol Astigarragak egindako lan handia.

5 Inplementazioa

Proiektu honetan poesia-sorkuntza automatikoaren egoera aztertu eta ikertzeaz gain, poesia-sortzaile bat garatu da.

Inplementazio lanek bi zutabe nagusi izan dituzte bertso-sortzaile automatikorako bidean. Batetik, egoera finituko teknologia erabilita egindako garapenak daude. Bestetik informazioaren berreskurapena eta semantika bateratzen dituen lanak.

5.1 Egoera finituko teknologia bertsolaritzan

Hasieran aipatu dugun bezala, aurretik garatuta geneukan lan bat egoera finituko teknologia erabilita berrinplementatu dugu. "Bertsotarako arbel digitala" proiektuaren lanetako bat errima-egiaztatzaile baten garapena izan zen. Aurten, masterreko irakasgai baten barruan, errima-egiaztatzailea *foma* konpilatzailea (Hulden, 2009) erabilita garatu dugu.

PHP lengoaiaz garatutako bertsioaren oinarria silabizazioa zen. Bertan, bi hitzek errimatzeko bete beharreko gutxieneko baldintza honakoa zen: bi hitzen azken silabak hoskideak izan behar ziren. Honetaz gain, azken aurreko silabetako bokala berdina bazen, orduan errimaren kalitatea ona zen, bestela pobre xamarra, baina egokia. Errima-egiaztatzailea eta errima-bilatzailea modu independentean garatu genituen, eta horren ondorioz, bilatzailearen emaitzetan agertzen ziren hitz batzuk, ez ziren onargarriak egiaztatzailearen arabera.

Hori ekiditeko, errima-egiaztatzailean eta bilatzailean teknologia bera erabiltzea erabaki genuen eta une honetan horrela daukagu garatuta, egoera finituko teknologia erabilita.

Errima-egiaztatzaile berriaren oinarria errima-patroiak dira. Honen oinarrizko *script*-a 11 irudian ikus daiteke. Lehenik eta behin, sarrerako bi hitzen **(landa-ganba)** errima-patroia identifikatzen da, eta gainerako guztia ezabatu (`rhympat` erregela). Errima-patroiak erauzitakoan **(anda-anba)**, bertsotan erabiltzen diren erregela fonetikoak aplikatzen zaizkie hauei (`phoRules` erregela). Azken hauek, ordezkapen erregela batzuk dira eta sarrera gisa "p", "t" edo "k" letrak dituen zerbait jasoz gero, adibidez, "PTK" karaktere-katearekin ordezkatuko dugu **(<aNMBDGRa>-<aNMBDGRa>)**. Honi esker, hoskideak diren kontsonanteen abstrakzio bat egingo dugu. Amaitzeko, *foma*-n erabilgarri dagoen `_eq(X, Left, Right)` funtzioa erabiltzen da. Funtzio honek, X tranduktorea aplikatzen dio sarrerako *string*-ari. Emaitza horrek, *Left* eta *Right* elementuez (kasu honetan < eta >) bereizitako bi kate izango ditu, gure kasuan errima-patroi bakoitza. Horiek berdinak baldin badira, kate berbera itzuliko du, bestela ez du onartuko **(gure adibidean, aNMBDGRa agertzen denez alde bietan onartu egingo du)**. 12 irudian errima-egiaztatzailearen pausoen eskema bat ikus daiteke.

Bertsolaritzan orokorrean, silabak kontatzen eta errimak egiaztatzen nahiko filosofia

```
define rhympat1   [0:"{" ?* 0:"}"
  [[[V+ C+]  (V) V] | [(C) V V]] C* ];
# constraining V V C pattern
define rhympat2  ~[?* V "}" V C];
# cleaning non-rhyme part
define rhympat3  "{" ?* "}" -> 0;
define rhympat rhympat1 .o. rhympat2 .o.
  rhympat3;

# rhyming pattern on each word
# and phonological changes
define MarkPattern rhympat .o.
       phoRules  .o. patroiak;
# verifying if elements between < and >
# are equal
define   MarkTwoPatterns
    0:%< MarkPattern 0:%> %-
    0:%< MarkPattern 0:%> ;
define Verify  _eq(MarkTwoPatterns, %<, %>)
regex Verify .o. Clean;
```

```
## LETRA MULTZOAK
define bdgr [b|d|g|r];
define ptk [p|t|k];
define nm [n|m];
define szx [s|z|x];



##ERREGELA FONETIKOAK
define petaka ptk -> PTK;
define bodegara bdgr -> BDGR;
define nomo nm -> NM;
define txistukari szx -> SZX;
define rt r PTK | s PTK -> t || _ .#.;
define ts PTK -> 0 || _ SZX;

define phoRules petaka .o.
bodegara .o. nomo .o. txistukari .o.
rt .o. ts;
```

11 irudia: Errima-egiaztatzailea *foma* erabilita.

hertsia jarraitzen bada ere, badaude estrofetan behar baina silaba gehiago sartzeko teknikak. Teknika hauek poesia klasikoan erabilitakoak dira, hizkuntz ezberdinetan. Horietako bi aurkeztuko dira hemen, *foma* formalismoan tratatuak izan direnak (hauek ere orain artean PHP lengoaian tratatzen ziren).

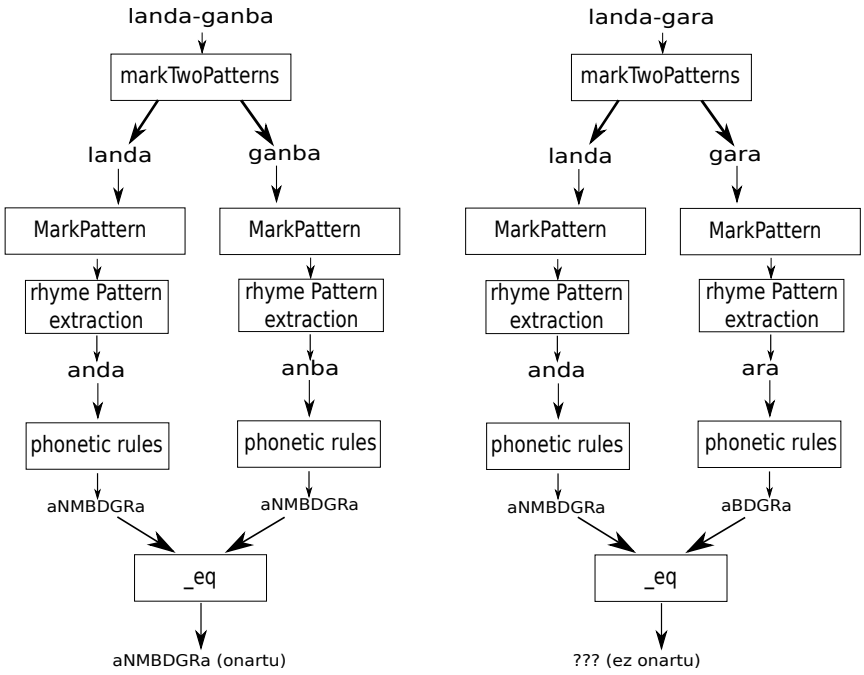
Lehenik eta behin, sinalefak dauzkagu. Sinalefa bat hitz bat bokalez amaitzen denean eta hurrengoa bokalez (edo “h” mutuaz) hasten denean gertatzen da. Sinalefa bat egiteko bi hitzek bertsoko lerro berean egon behar dute, hau da, lerro bateko azken hitza bokalez amaitzen bada eta hurrengo lerroko lehena bokalez hasi, ezin da sinalefarik egin.

Honetaz aparte, [bokal + “h” + bokal] sinkopa patroia daukagu, betiere, bokalak berdinak badira. Patroi hori topatuz gero, bokal bat soilik uzteko aukera dago, eta ondorioz, bi silaba zeuden tokian bat bakarrik kontatzen da.

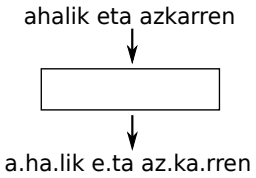
Aurretik esan bezala, hitzezko figura erretoriko hauek maila konputazionalean tratatuta zeuden PHP lengoaia erabilita. Ondorengo lerroetan *foma* bidez egindako inplementazioaren nondik norakoak emango ditugu.

Figura hauek hitzen silabizazioari eragiten diote, beraz, hau silabizazio arruntaren ondorengo atal bat izango da. Egoera finituko silaba-banatzaile arruntak hitz bat edo esaldi bat hartu eta silabetan banatzen du:

HAP masterra



12 irudia: Errima-egiaztatzailearen fluxuaren adibidea.



13 irudia: Silaba banatzaile sinplea

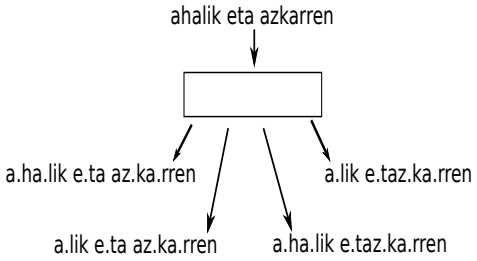
Bi figura erretoriko hauek kontrolatzeko, *foma*-ko silabizazio kodeari ondorengo lerro hauek txertatu behar izan dizkiogu.

```
define SPhSP " " (->) 0 || V _ V ;

define aha a "." h a (->) a ;
define ehe e "." h e (->) e ;
define ihi i "." h i (->) i ;
define oho o "." h o (->) o ;
define uhu u "." h u (->) u ;
define vhv aha .o. ehe .o. ihi .o. oho .o. uhu ;

define flex vhv .o. SPhSP ;
```

Argi ikus daitekeenez, `SPhSP` transduktoreak hutsunea kentzen du, alde bietako hitzek bokalez inguratzen badute. Hau da, “etxera etorri” hitza sartuko bagenu, hutsunea ken-

duko luke. `vhv`-ren kasuan, [bokal + "h" + bokal] patroia bilatzen du, betiere, bokalak berdinak izanik, eta hau bokal sinple batekin ordezkatzen du.

Bi transduktore hauek, baina, badute berezitasun bat: ordezkapen sinboloa hautazkoa da [`(->)`]. Honekin zera esan nahi da, hautazkoa dela malgutasun teknika aplikatzea. Horri esker, esaldi baten silabizazioa eskatzean sistemari, aukera posible guztiak itzuliko dizkigu honek:



14 irudia: Silaba banatzailea malgutasunarekin

5.2 Bilatzaile semantikoa

Arantxa Otegiren lanari jarraiki (Otegi, 2012), informazioaren berreskurapeneko[26] teknika berritzaileenak erabilita, bilatzaile semantiko bat garatu dugu. Honi esker, hitz bat gaitzat emanda, bertso batekin erantzuteko aukera izan dugu.

Baina nola jakin zein den hitz horri hobekien egokitzen zaion bertsoa? Bertsoen corpus bat dugu eskura, beraz, bilaketak horren gainean egin ditzakegu. Baina zein bilaketa litzateke egokiena? Suposa dezagun "bizikleta" hitzaren inguruko bertsoak bilatu nahi ditugula. Maila baxueneko bilaketak, bertsoetan "bizikleta" *string*-a bilatuko luke. Kasu horretan, agian zerbait topa dezakegu, baina imajinatu bertso batean honako estrofa hau dagoela:

Atzo ibili ginen lagunak
etxeko bizikletekin

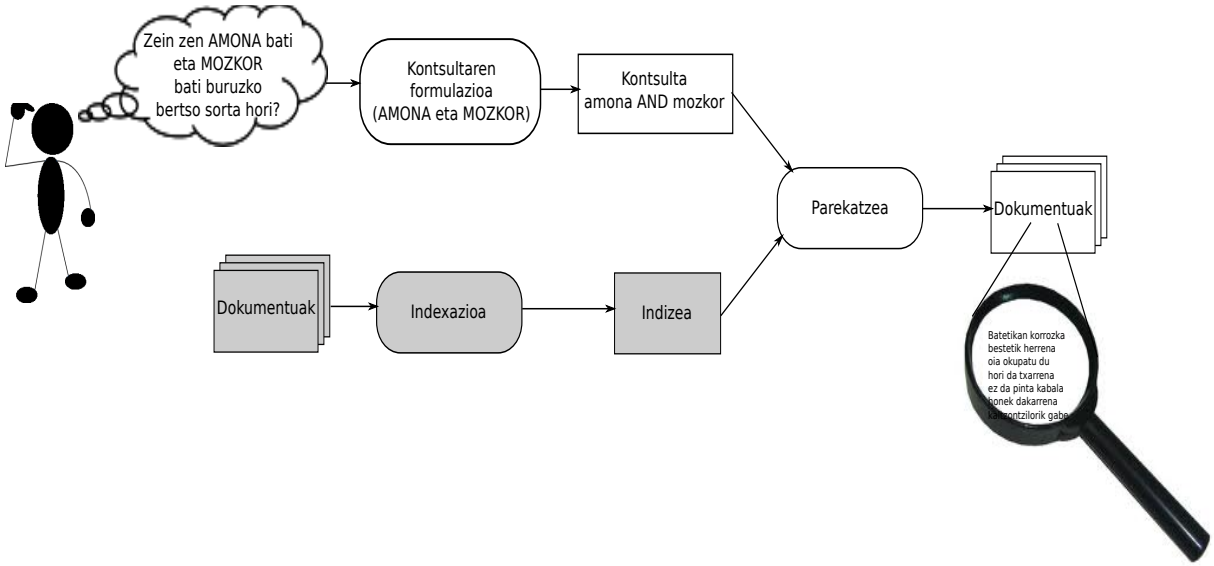
Bertso hori berreskuratzea interesgarria izan daitekeen arren, ***string*-en parekatze soilak** erabiltzen baditugu ezingo dugu horrelako bertsorik topatu. Beraz, estaldura hobetzeko, bertsoen gainean bilaketak egitean, bertsoko hitzen **lemen gainean** ere egingo dira bilaketa horiek. Bigarren kasu hipotetiko batean, pentsa bertso baten amaierak hau dioela:

[26] Information Retrieval

gogoan daukat lehen aldia
txirrinduaren gainean

Kasu honetan ere, oso erlazionatua dago “bizikleta” hitzarekin. Baina, karaktere-kateak ezberdinak direnez bertso hau ezingo genuke berreskuratu. Zer egin daiteke bertso hau itzultzeko? Euskal Wordnet bezalako ezagutza-base lexikal bat (Pociello, 2008) erabilita, bertsoko lemen sinonimoak, hiperonimoak, hiponimoak, etab. lortuko ditugu (**hedapen semantikoak**), eta gero horien gainean ere bilaketak egin.

Bilaketak azkarrak izan daitezen, bertsoetako hitzak, hitzen lemak eta haien hedapen semantikoak indexatu egin ditugu. Lematizazioak, hedapen semantiko guztiak eta indexatzeak egiteko, **Arantxa Otegi**k bere doktorego-tesirako garatutako *script* batzuk erabili ditugu. Horiei aldaketa txiki batzuk eginda, bertsoen berreskurapenerako erabiltzea lan erraza izan da. 5.2 irudian informazio-berreskurapeneko tresnen egitura nagusia ageri da, baina bertsolaritzaren inguruko adibide batekin.



15 irudia: Bertsoen bilatzaile semantikoaren egitura

Dokumentuen indexatzea hiru pausotan egiten da. Lehenengo pausoan indexatu beharreko dokumentu zerrenda eskuratzen da.

```
find bertsoenDirektorioa -type f > bertsoFitxZerrenda.files
```

Behin dokumentu zerrenda hau dugula, berau `collection` motako fitxategi batean sartzeko ordua da. Pauso honetan dokumentuek izango duten egitura ere definitu egin

behar da. Hori definitzeko *mg4j* sistemak `myFactory` izeneko klase bat dauka. Badaude, horretaz gain, aurredefinitutako klase batzuk, `HtmlDocumentFactory`, adibidez. Gure dokumentuek, zazpi eremutako egitura bat daukate. Hona hemen eremu bakoitzean gordetzen den informazioa:

- 0. eremua: Bertsoaldiaren identifikadorea, bilduma eta zenbaki bat. Adibidez, 86_1 ("Bapatean 86" bilduma eta 1 zenbakidun bertsoa).
- 1. eremua: Bertsoaldiaren gaia
- 2. eremua: Bertsoaldiaren transkripzioa
- 3. eremua: Bertsoaldiaren gaiaren lemak
- 4. eremua: Bertsoaldiaren transkripzioaren lemak
- 5. eremua: Bertsoaldiaren gaiaren hedapen semantikoak
- 6. eremua: Bertsoaldiaren transkripzioaren hedapen semantikoak

Ondorengo aginduak dokumentu zerrenda bat (`bertsoFitxZerrenda.files`) eta hauen egitura (`myFactory`) eskuratu eta `collection` motako fitxategi bat (`bilduma.collection`) itzultzen du.

```
java it.unimi.dsi.mg4j.document.FileSetDocumentCollection -f myFactory
-p encoding=UTF-8 bilduma.collection < bertsoFitxZerrenda.files
```

Azkenik, bilatzaileak modu azkarrean aurkitu ahal izateko, indizea sortzeko ordua da. Horretarako, `IndexBuilder` klasea erabili behar da eta honi, honako datu hauek adierazi behar zaizkio: indexatuko diren eremuak, `collection` fitxategia eta gordeko den indizearen oinarrizko izena.

```
java -Xmx2058M it.unimi.dsi.mg4j.tool.IndexBuilder --downcase
-I f0 -I f1 -I f2 -I f3 -I f4 -I f5 -I f6 -s 10000 -S bilduma.collection
indize
```

Behin dokumentuak indexatuta ditugunean bilaketak egitea besterik ez zaitu geratzen. Horretarako, *mg4j*-ko `Query` klasea erabili behar da. Honi esker, bilatzaile semantikoa martxan jartzen da eta web bidez edo terminal bidez egin daitezke kontsultak.

```
java it.unimi.dsi.mg4j.query.Query -h -i FileSystemItem -c bilduma.collection
indize-f1 indize-f2 indize-f3 indize-f4 indize-f5 indize-f6
```

HAP masterra

Ebaluazioa

Informazio-berreskurapenaren inguruan, modu ezberdinak daude bilatzaileak ebaluatzeko. Guk, bilatzailearen doitasuna neurtu dugu, honetarako bi neurri ezberdin erabili ditugularik:

Precision at N

Bilaketaren lehen N emaitzak hartu eta esanguratsuak diren ala ez ikusi. Horren arabera, puntuazio bat emango zaiolarik:

$$Precision\ at\ N = \frac{\sum_{pos=1}^{N}(esanguratsu(pos))}{N};$$

esanguratsu(pos) funtzioak 1 edo 0 itzuliko du, berreskuratutako dokumentua esanguratsua den ala ez. Berreskuratutako dokumentu kopurua finkatutako N zenbakia baina baxuagoa bada, N berreskuratutako dokumentu kopurua izango da.

Gure ebaluazioetarako, berreskuratutako lehen 5 dokumentuak erabiliko ditugu.

Average Precision

AP neurriak, berreskuratutako dokumentu adierazgarri guztien posizioei dagokien doitasunaren batezbestekoa kalkulatzen du (Otegi, 2012, 66 orr.). Neurri hau erabiltzea interesgarria da gure bilaketa-motoreak (*mg4j*) emaitzak *ranking* batean ordenatzen dituelako emaitzak eta neurri honek *ranking* hori ere kontutan hartzen du. Kasu honetan ere, lehen bost emaitzak bakarrik izan ditugu kontuan gure ebaluazioan.

$$AP = \frac{\sum_{pos=1}^{N}(P@pos \times adierazgarri(pos))}{dokumentu\_adierazgarrien\_kopurua}$$

Ebaluazioa egiterako orduan, berau garatzailea ez den norbaitek egitea ezinbestekoa da, beraz, horretarako, Bertol Arrietaren laguntza izan dugu. Bertolek 24 bilaketa termino prestatu ditu eta gero lehen bost emaitzen esangura ebaluatu. Hauetaz aparte, bi hitz edo gehiagoz osatutako bilaketa-termino batzuk ere bilatu ditu. Ondorengo taulan daude ebaluazioaren emaitzak:

Hitz kop.	Test kop.	Pr. at 5	MAP
1	24	0.7625	0.3813
2	12	0.5166	0.3336
3	5	0.1333	0.0889

HAP masterra

Oraintxe aipatu bezala, hitz bateko 24 bilaketa egin ditu Bertolek bilatzaile semantikoan. Argi erakusten dute emaitzek, batez ere *Precision at 5* neurrikoek, hitz bat baino gehiagoz osatutako bilaketa terminoekin emaitza kaxkarragoak lortzen direla. Aipatzekoa da baita bilatutako hitzen kategoria: izen arruntak, adjektiboak eta aditzak. Orokorrean izenekin emaitza hobeak ematen ditu, gehienbat Euskal WordNet-en aditz eta adjektiboak baino gehiago daudelako.

6 BertsoBOT: Zer egiteko gai da?

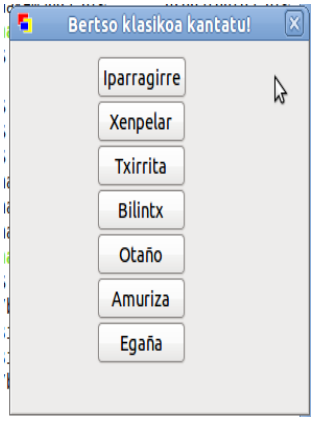
BertsoBOT da bertsotan egin dezakeen sistemaren izena. BertsoBOT, sarrera ezberdinak jasota, pertsonekin bertsoen bidez komunikatzeko gai da.

Bere jarduna, hainbat mailatan erakuts daiteke. Ondorengo azpiataletan, bere gaitasunak deskribatuko dira, maila baxuenetik altuenera:

6.1 Memoria hutseko lana: bertso klasikoa

Lehen maila, memoria hutseko lana da. Kasu honetan, hainbat bertso klasikoz osatutako datu-base bat dauka sistemak. Bertso bat kantatzeko eskatzen zaionean, bere kutxa sinple honetatik bertso bat ausaz atera eta hau kantatzen du, besterik gabe. Bertso-sorkuntza mailan, honek ez du inolako meriturik, lehendik sortutako bertso bat aukeratu eta kantatu egiten duelako, inongo aukeraketa irizpiderik gabe.

Ariketa hau egiteko, perl programazio-lengoaia erabilita, interfaze grafiko bat garatu genuen. Interfaze hau informatika fakultatean egin zen FLLXpress lehiaketan erabili zen, Tartalo robotarekin. 16 irudian programaren leiho nagusia ikus daiteke, non erabiltzaileak bertsolari bat aukeratu behar duen. Behin hori aukeratuta, haren errepertorioko bertso bat ausaz kantatzen dut robotak.



16 irudia: Bertso klasikoen ariketa egiteko sortutako interfazea.

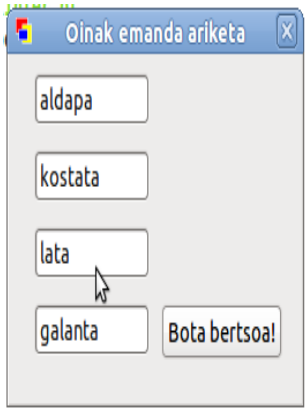
6.2 Corpusen gaineko bilaketa arrunta: lau oinak emanda

Bigarren maila honetan, bilaketa arruntak erabiltzen ditu bere jarduna burutzeko. Bilaketa sinple hauei esker, lau oinak jasota, estrofez osatutako corpusean bilaketak egiten ditu. Oin

horiei dagozkien estrofak uztartu eta bertso bat itzultzen du.

Kasu honetan, bertso-sorkuntza lan bat egiten da. Lau hitz emanda, oin horiek errespetatzen dituen bertso bat lortzen da. 4.1.2 atalean azaltzen den bezala, posible da oin batekin estrofarik ez aurkitzea. Hori gertatuz gero, "la" silabez betetzen da estrofa eta bertsoBOT-ek bertsoa botatzen du.

Aurreko ariketarekin bezala, honentzako ere interfaze grafiko bat inplementatu genuen, FLLXpress lehiaketaren aitzakiarekin. 17 irudian ikus daiteke lau oinak idazteko leihoa eta bertan agertzen den "Bota bertsoa!" botoia sakatuta, bertsoa kantatzen du Tartalok.



17 irudia: Oinak emanda bertsoa kantatzeko sortutako interfazea.

6.3 Corpusen gaineko bilaketa adimenduna: hitza emanda

Bilaketa adimenduna egiteko, 5.2 atalean azaldutako bilatzaile semantikoa erabili da. Lehenago esan bezala, honi esker, hitz bat emanda, sistema bertso bat itzultzeko gai da. Beraz, bertso-saioetan ohikoa den gai bati erantzuteko prest dagoela esan genezake. Ebaluazioaren atalean, bilatzaile semantikoaren beraren ebaluazio bat ere aipatzen da, beharrezkoa ikusi baita hau bere horretan ebaluatzea.

Bilatzaile semantikoak, *mg4j* bilaketa motorea erabiltzen du. Arantxa Otegik prestatutako kodetik abiatu gara. Kasu batzuetan *java* kodeari aldaketa txiki batzuk ere egin behar izan dizkiogu, terminal bidez berreskuratutako bertsoak jaso ahal izateko, adibidez.

6.4 Bertso-sorkuntza automatikoa: bertsoa gai librean

BertsoBOT-ek burutu dezakeen azken ariketa hau, oraindik esperimentala da. Egia da bertso-sorkuntza egiten duela, hau da, gai librean, unean uneko bertso bat botatzeko gai

dela. Honek duen arazoa zera da, ez duela perpausik osatzen, baizik eta hitzak (izen, adjektibo, aditz ...) ausaz gehitzen ditu. Metrika aldetik, artearen egoerako beste lan batzuen gisara, bertso perfektuak sortzen ditu. Erabilitako errimak ere errima onargarriak dira, (Amuriza, 1981b) lanaren arabera[27].

Etorkizuneko gure helburua hemen sortutako bertsoak koherente eta zentzudunagoak izatea da, bai gramatikaltasun aldetik, bai pragmatikaren aldetik, eta baita poetikotasunaren aldetik ere. Azpian ageri den bertsoa, "Iparragirre abila dela" bertsoaren neurrian dago eginda.

hurbildu lehen naiz gutxiago
gera hiru badakizu
giro bela korrika luzaro
benetan jan dagokizu
zuk gainetik dugula
bera zuk txapitula
barruan prest dakarzkizu
gerora orduan beharko bi
ein baina itzuli pisu

Ebaluazioa

BertsoBOT-en ebaluazioa burutzeko bide ezberdinak daude, hala erakutsi digute artearen egoeran dauden lan batzuek.

Gure kasuan, ebaluazio bat, robot eta giza bertsolarien arteko bertso-saioak izan zuen harrera ona izango litzateke. Robotak saioan kantatu zituen bertsoak[28] egokiak izan ziren, hau da, orokorrean, lekuz kanpoko gauzarik ez zuen egin.

- Bertso klasikoa abesterakoan, aipagarria da, bertsoren bat azkarregi kantatu zuela. Hori helarazi ziguten entzule batzuek. Honek ez digu asko ardura, izan ere, kantatze-abiadura aldagarria da.
- Oinak emanda bertsoa abesteko ariketaren kasuan, hasieran robota ez zen bertsoa botatzeko gai izan. Kontua da, bi oin baino gutxiago aurkituz gero corpusean, bertsorik ez kantatzea erabaki genuen, arraroa baita lau estrofako bertso batean "la"-z betetako hiru estrofa entzutea. Edozein hitz emanda, erantzuteko estrofa bat izateko, une honetan daukagun zortziko txikien corpusa berrelikatu eta handitzea da soluzioa.

[27] "Bertsotarako arbel digitala" proiektuko errima-bilatzailea erabiltzen du.
[28] Eranskinetako B atalean bertso hauen transkripzioak daude ikusgai.

- Hirugarren ariketan, hitza emanda bertsoa kantatzerakoan, konexio arazo bat izan genuen. Arazo hori gure garapenetik at gertatu zen, hau da, ez zen gure kontua izan. Baina beste aukera bat eman zitzaionean, ariketa gisa, "ikasle" hitza eman zitzaion eta bertso egoki bat bota zuen.
- Hutsetik sortutako bertsoaren kasuan, ez zen arazorik egon. Bertsoa ez zuen inork ere ulertu, baina oraindik modu esperimentalean jarri genuen ariketa hori, aurrez jakinda zentzu gutxiko zerbait kantatuko zuela.

7 Ondorioak eta etorkizuneko lana

Lan honetan, bertso- edo poesia-sorkuntza automatikoan lehen urratsak eman dira. Aurretik egindako beste lanetan eta informazioaren-berreskurapenean oinarrituta, bertsoen corpus batetik hauek berreskuratzeko bilatzaile semantiko bat ere egin da. Corpus hau da, TEI5 etiketatze-araudi estandarra erabilita etiketatu dena.

Gure lanak hiru modulutako arkitektura bati jarraituz egin ditugu, bertso-sorkuntza atala, testu-kantu bihurketa atala eta mugimenduen atala. Hemen aurkeztu den lanaren muina, lehen atalean dago, hau da, guk bertsoen sorkuntza automatikoan egin (eta egingo) dugu enfasia.

Horretaz aparte, BertsoBot-en lau gaitasun-maila ezberdin ere plazaratu ditugu lan honetan: Bertso klasikoa kantatu, oinak emanda bertsoa kantatu, hitza emanda bertsoa kantatu eta hutsetik sortutako bertsoa kantatu. Gainera, aurretik egindako lan bat –errima-egiaztatzailea– berriro inplementatu da, egoera finituko teknologia erabilita.

Etorkizunean lan asko dago egiteko, eta lan hauen ondorio izango dira garatuko diren bi doktorego-tesiak. Hona hemen lanetako batzuk:

- Egoera finituko teknologien haritik, oraindik errima-egiaztatzailea fintzeko daukagu. Adibidez, “errima” eta “filma” hitzek, errima-patroi ezberdinak dituztenez, ez dute errimatzen gure azken egiaztatzailearen arabera. Kasu horretan, “l” soinua, hitz horretan oso garrantzitsua ez denez, bi hitz horiek errimatzen dutela esan genezake. Beraz, **soinu edo fonema ez garrantzitsu horien detekzioa**, errima-egiaztatzailearen baitan, izango litzateke etorkizuneko lanetako bat.
- Urte honetan semantikaren inguruan urrats batzuk eman diren arren, etorkizunean **sintaxi eta pragmatika jorratzeko** intentzioa dugu. Denok dakigu, bertso eta poemek gramatikaltasun eta metrika hertsitik haratago, esanahi bat eta munduari buruzko jakinduria dutela. Beraz, hauek maila konputazionalean aztertu nahi ditugu, eremu horretako tesi ezberdinak aztertu eta bateratuaz. Baina, semantikaren inguruan aurten lan egin izanak ez du esan nahi bere bide guztiak agortutzat eman ditzakegunik. **Semantikaren adarrek oraindik zabalik** darraite eta horiek ere aztertuko ditugu.
- **Ikasketa automatiko**ko teknika berritzaileenak erabiliko ditugu. Ikasketa automatikoa, artearen egoerako lan gehienek erabiltzen duten teknika da, azken urteotan gorakada nabarmena izan duena. Ikasketarako, hainbat atributu ezberdin erabiliko ditugu, bertsoen corpusetako patroi morfosintaktikoak, bertso bakoitzeko izen, aditz edo adjektibo kopurua ...
- Bertsolaritza gure zutabe nagusietako bat izanagatik ere, ez ditugu **bestelako adierazpen literario batzuk** alboratuko, hala nola, poesia arrunta, haikuak, ipuinak, etab.

- IXA taldean garatu diren eta artearen egoeran dauden lanen ildotik (Aldabe et al., 2006), **bertsolaritzaren inguruko ariketak automatikoki sortzeko** sistema baten garapena da beste aukera bat. Hau bertso-eskolekin elkarlan estuan egingo genuke.

- ***Bertso-playground* jokua** sortzeko intentzioa dugu, non bertso bat erabiltzaile ezberdinen artean osa daitekeen eta honen arabera puntuazio ezberdinak eskuratuko dituen erabiltzaile bakoitzak.

- Hizkuntzari dagokionez, nagusiki euskararen inguruan lan egingo dugu, baina, gure helburu nagusia, testu poetikoak idazteko gai izango den **sistema eleanitz bat sortzea** da. Honetarako, garrantzi handia izango du aurtengo uztailean Donostian egin zen FSMNLP[29] kongresuan aurkeztutako (Novak et al., 2012) lanak.

- Goian aipatu diren lan guztiei, **robotika lanak** txertatu behar zaizkio. Honetarako, Robotika eta Sistema Autonomoak taldearekin batera egingo ditugu lan asko, eta bereziki, Aitzol Astigarragarekin.

[29] Finite-State Methods And Natural Language Processing

Eranskinak

A Kasu bidezko arrazoitzea

Kasu bidezko arrazoitzea, orotariko problemak ebazteko metodo bat da, zeina aurretik ebatzitako problemetan oinarritzen den. Lau urrats ditu kasu bidezko arrazoitzeak, **berreskurapena**, **berrerabilpena**, **berrikuspena** eta **gordetzea**.

- Berreskurapena: Arazo bat emanda, memorian gordeta dauden antzeko kasu bat berreskuratu. Kasu baten elementuak honako hauek dira: arazoa, honen ebazpena eta beharrezkoa balitz soluziora iristeko oharrak.
- Berrerabilpena: Berreskuratutako kasua uneko beharretarako moldatzen da pauso honetan.
- Berrikuspena: Behin soluzio bat izanda, mundu errealean soluzio hau probatu ea onargarria den eta beharrezkoa balitz, zuzendu.
- Gordetzea: Ebazpena gure arazoarentzako modu egokian moldatu ondoren, hau kasu berri modura gorde etorkizuneko arazoetarako lagungarri izango delakoan.

Adibidea [30]

Adibide honetako protagonistak, Fred-ek, ahablazko (*arándano*) krepeak egin nahi ditu, baina zoritxarrez, ez daki hauek prestatzen, sukaldari hasiberria delako. Badaki, ordea, krepe arruntak egiten. Krepe arruntak egiteko metodoa izango da **berreskuratuko** den kasua. Krepeak ahablekin egin nahi dituenez, nolabait aldatu egin beharko du errezeta, nonbait fruituak gehitzeko, hau **berrerabilpen** atala izango da. Suposatuz, ahablak ore bustiaren gainera bota dituela, Fred-i ez zaio gustatu oreak hartu duen kolore urdina, beraz, **berrikuspen** atalean, fruituak gehitzeko orea lehortzea itxarotea proposatuko du. Behin hau eginda, Fred-ek bere errezeta berria **gorde** egingo du, eta horrela hurrengoan jakingo du ahablazko krepeak egiten.

[30] Iturria: English Wikipedia

B Robot eta giza-bertsolarien bertso-saioa

AGURRA

Galtxagorri	**Tartalo**
Egunon denei hementxe gatoz	Ohitura denez hasi beharko
dotore eta galako	Aupa! Zer moduz, esanez
Galtxagorri ta neretzat ere	lau bertsolari eta bi robot
lehenengo plaza dalako	talde xelebrea zinez
begiak xuxen, kableak txukun	ahalik eta saio onena
burua programez lepo	josteko itxaropenez
gure partetik dibertsiorik	esperimentu, berrikuntzekin
entzule ez da faltako	sarri gertatzen den legez
barruan degun bertsolaria	nola hasi den denok dakigu
aterako degulako	nola amaituko den ez (bis)

BERTSO KLASIKOA

Galtxagorri	**Oier Lakuntza**
Iparragirre abila dela	Entzun dut eta guztiz harritu
askori diot aditzen	ezin zen izan bestela
eskola ona eta musika	sekula ez zen jardun robot bat
hori hoiekin zerbitzen	gaur Galtxagorri bezela
ni ez nauzu ibiltzen	hemen gabiltz kantari
kantuz dirua biltzen	gu gaur sei bertsolari
komediante moduan	konturatu naiz honela
debalde festa preparatzen det	Iparragirre ez dakit baina
gogua dedan orduan	hauxe bai abila dela

Galtxagorri	**Andoni Egaña**
Modu horretan jarri zuenak	Hara Txirritan bertso zahar hoiek
norteko ferrokarrila	bota ditu azkenian
jakinduria handia zuen	ferrokarrila hor aipatzen zun
entendimendu abila	progreso handi planian
gizonik ezin hasi liteke	orain Irundik Madrila goaz
zaldi hobearen bila	lau orduan gehienian
goizean irten Irundik eta	hala zer gauza genitun leno
iluntzerako Madrila	robotikan ez zanian

HAP masterra

Galtxagorri	**Oier Lakuntza**
Loriak udan ihintza bezela	Bere garaian Bilintxek zuen
maite det dama gazte bat	hain bertso ederra hau ta
hari hainbeste maite dionik	gureraino re ailegatu da
ez da munduan beste bat	hori ez daiteke uka
inoiz edo behin pasatzen badet	ni robotekin harriturikan
ikusi gabe aste bat	nago baina diot aupa
bihotz guztira pasatutzen zait	tristura nuke honen mailara
halako gauza triste bat	aritzerik ez baneuka

Galtxagorri	**Andoni Egaña**
Mendian gora haritza	Galtxagorri hitzen mago
ahuntzak haitzetan dabiltza	ez da sentitzen naufrago
itsasoaren arimak dakar	robotak etzun maila hau eman
ur gainean bitsa	sekula lehenago
kantatu nahi dut bizitza	ze intentzio, ze gogo
usteltzen ez bazait hitza	hiztegitik zenbat trago
munduan dantzan jarriko nuke	halare inoiz etzera robot
jainkoa banintza	jainkoa izango

HITZA EMANDA

HITZA: LANGABEZIA

Felix Zubia	**Galtxagorri**
asko nahi izan gabe	-
dijuaz kalera	-
tailerrak itxi eta	-
pertsianak behera	-
ipuin honek nun dauka	-
bere bukaera	-
hortxe hasten ote da	-
gazteen karrera	-
ipuin honek nun dauka	-
bere bukaera	-
unibertsitatetik	-
saiatuko gera	-

HITZA: IKASLE

Felix Zubia	**Galtxagorri**
mundura sortu ginen	azken boladan nola zabiltzan
modu bihusian	suspentsotik suspentsora
ezjakinak gerala	espabilatu zaitez gaztea
ondo ikusian	alde daukazu denbora
orain irakasle naiz	ahaztu nahi det baina ezin ahaztu
erdi nagusian	nola igo zeran gora
nahiz ta nere burua	azken aholku bat det zuretzat
erdizka ikusian	eta ekarri gogora
orain irakasle naiz	biok berdinak gera bai baina
erdi nagusian	ikasgelatik kanpora
ikasle izateko	
bizitza guztian	

OINAK EMANDA

Maddalen Velarde	**Galtxagorri**
nahiz ta gutxitan garen	Sumatu genezake
gazte eta **aske**	hainbat **bertsolari**
nik oraindik luzaro	esperantzaz gabiltza
izan nahi det **gazte**	gaurkoan **kantari**
nahiz helduentzat dugun	hoiek ere ez ditut
beti eske ta **eske**	izango **ugari**
hori baino gehiago	euroa hemen da ta
gu izan **gaitezke**	egon zaitez **adi**

Tartalo	**Maddalen Velarde**
-	tartalok kantatzeai
-	egin dio **boikot**
-	biok nahasten gara
-	gizaki ta **robot**
-	bigarren aukera bat
-	behar dula **diot**
-	guztiok berdin berdin
-	ein dezagun **porrot**

HAP masterra

GAI LIBREAN

Tartalo	**Galtxagorri**
adina erdia joan	hobea falta idatzi
ene negarra aintzintoan	arrazoi nauzu amatxi
oraindik danari estiballes	lagun edo herri gaitezen ta
hainbeste zeroan	uranga solatxi
daukat gindoazkioaan	haundia zeuk eskuidatzi
danama esker neroan	ulertu geran ba patxi
jo zoriona muga naizela	guztiak ziren dauzkat gera hein
hobea zegoan	duena jan patxi

C Proiektu honen harira argitaratutako artikuluak

Proiektu honen eta aurrekoen ildotik, bi artikulu argitaratu ditugu:

- BAD: An assistant tool for making verses in Basque (Agirrezabal et al., 2012a)
- Finite-state technology in a verse making tool (Agirrezabal et al., 2012b)

BAD: An assistant tool for making verses in Basque

Manex Agirrezabal, Iñaki Alegria, Bertol Arrieta
University of the Basque Country (UPV/EHU)
maguirrezaba008@ikasle.ehu.es, i.alegria@ehu.es, bertol@ehu.es

Mans Hulden
Ikerbasque (Basque Science Foundation)
mhulden@email.arizona.edu

Abstract

We present work on a verse-composition assistant for composing, checking correctness of, and singing traditional Basque *bertsoak*—impromptu verses on particular themes. A performing *bertsolari*—a verse singer in the Basque Country—must adhere to strict rules that dictate the format and content of the verses sung. To help the aspiring *bertsolari*, we provide a tool that includes a web interface that is able to analyze, correct, provide suggestions and synonyms, and tentatively also sing (using text-to-speech synthesis) verses composed by the user.

1 Introduction

In the Basque Country there exists a long-standing live performance tradition of improvising verses—a type of ex tempore composition and singing called *bertsolaritza*. Verses in *bertsolaritza* can be seen as discourses with strict rules governing the technical structure of them: verses must contain a certain number of lines and each line must have a defined number of syllables, certain lines have to rhyme in certain patterns, and so forth.

In this paper we present a web-based assistant tool for constructing verses (*bertsoak*) according to the rules of *bertsolaritza* (Garzia et al, 2001).

If the reader is interested in this topic, we recommend watching the 2011 film *Bertsolari*[1] [2], directed by Asier Altuna.

[1]IMDB: http://www.imdb.com/title/tt2058583
[2]Trailer on: http://vimeo.com/9355066

2 Relationship to earlier work

There exist some prior works dealing with Basque verse-making and computer technologies, such as BertsolariXa (Arrieta et al., 2001), which is a rhyme search tool implemented as finite-state automata using the two-level morphology formalism. The tool also contains other features, including semantic categorization of words, narrowing word-searches to certain themes, etc. While BertsolariXa focuses mostly on the word-level, the current work also includes constraints on overall verse structure in its implementation as well as a synonym search tool, a melody suggestion system, and possibilities for plugging in text-to-speech synthesis of verses.

2.1 The *Bertsolari* tradition

Bertsolaritza is very ingrained in the Basque Country and championships, competitions and get-togethers on *bertsolaritza* are quite common. Usually the competitors in such event, called *bertsolaris*, are given a theme to produce a verse on under some very limited time constraints.

But the Basque Country is not the only place that hosts such troubadour traditions—similar customs are present in many other countries such as Cuba, Brazil, Argentina, etc. The goal of the current tool is to be generalizable, and so applicable to various strategies of verse improvisation, and possibly be useful not only for Basque speakers, but also for others.

Below we briefly present an example of a verse made in the Basque Country. In 1986 Andoni Egaña (a well-known *bertsolari*) was asked to sing a *bertso* and assigned a topic. In the verse, he was asked to play the role of an old person who lived alone, and who realized that he could

not even tie his shoes. Within a few seconds he composed and sang three verses. Here, we analyze the first verse.

Verse:
Gazte aroan ibili arren
gustora tirriki-tarra,
denbora honen joan etorriak
ederki jo dit gitarra,
gorputza daukat ximeldurikan
ta eskuen punta zaharra,
denborarekin seko galdu det
gazte aroko indarra,
ez al da pena gizon mardul bat
hola ibili beharra.

Translation:
Even when I was young
I was always on a spree
over time
I have been punished
I have a crumpled body
and the tip of the hands very old,
Over time I lost
the strength I had when I was young,
It's a shame that a strong man
has to end up like me.

The special charm of *bertsolaritza* improvisation is that people proficient in the art can quickly express a variety of ideas, although they are working with very restrictive rules concerning the number of syllables in words they use, and how the words must rhyme. We must take into account that Andoni Egaña was able to sing this verse within a few seconds of being given the topic, and also, that it complies exactly with a certain metric. In this case, the verse contains eight lines, each odd line consisting of ten syllables, and each even line of eight syllables, with the even lines rhyming.

Formal training in the *bertsolari* tradition also exists in the Basque Country. In the last 20 to 30 years, an important movement has developed that aims to provide instruction to upcoming generations on how to create verses (orally or in writing). This kind of instruction usually takes place in learning centers called *bertso-eskolak*, which in English roughly means, "verse-making schools." The proliferation of this movement has produced a strong base of young *bertsolaris*, of whom many achieve an outstanding level of improvisation skills.

3 The BAD tool

BAD is the acronym for "*Bertsotarako Arbel Digitala*", roughly "Digital verse board." The aim of the tool is to serve as a general assistant for *bertsolari*-style verse composition and help verse-making learners in their learning process.

This tool has been developed using the PHP programming language, but it contains certain parts developed using finite-state technology. The main functions of this tool, which will be discussed in more detail in the next five sections, are the following: visualization of the verse structure, structure checking, rhyme and synonym searching and verse singing.

3.1 Verse structure

The main rules of the *bertsolari* verse are that a verse must consist of a certain predefined number of lines and each line in turn, of a predefined number of syllables. Traditionally, about a hundred different schemes are used, and the tool provides support for all these patterns. For example, the structure called "*Hamarreko handia*" has ten lines and ten syllables in the odd-numbered lines, and eight syllables in the even-numbered lines. In this structure, the even-numbered lines have to rhyme. Selecting this scheme, the tool will mark the corresponding lines with their requirements.

The web interface can be seen in figure 1, which shows the general layout of the tool, illustrated with the example verse referred to above—we see that each line has been approved in terms of line length and syllable structure by the tool.

We have designed a database in which the main verse structures are saved so that when the user selects one verse schema, the system knows exactly the number of lines it must contain, where must it rhyme and how many syllables each line should have. Those schemata are also linked to melodies, each melody corresponding to one possible structure.

3.2 Structure checking

After writing the verse, the system can evaluate if it is technically correct, i.e. if the overall structure is correct and if each line in the form abides by the required syllable count and rhyming scheme. The syllable counter is implemented using the *foma* software (Hulden, 2009), and the implementation (Hulden, 2006) can be found on the homepage of

Figure 1: A verse written in the BAD web application.

foma.[3]

Separately, we have also developed a rhyme checker, which extracts special patterns in the lines that must rhyme and checks their conformity.

These patterns are extracted using *foma* (see section 3.4) after which some phonological rules are applied. For example, an example rule *era* → {era, eda, ega, eba}, models the fact that any word ending in *era*, for example, *etxera*, will rhyme with all words that end in *era*, *eda*, *eba* or *ega*. These rhyming patterns have been extracted according to the phonological laws described in (Amuriza, 1981).

3.3 Synonym search

Usually, people who write verses tend to quickly exhaust their vocabulary and ideas with to express what they want to say, or encounter problems with the number of syllables in various tentative words they have in mind. For example, if the verse-maker wants to say something containing the word "family," (*familia* in Euskera, a four-syllable word) but is forced to use a three-syllable word in a particular context, the interface provides for possibilities to look for three-syllable synonyms of the word *familia*, producing the word *sendia*— a word whose meaning is otherwise the same, and made up of three syllables.

For developing the synonym search, we used a modified version of the Basque Wordnet (Pociello et al., 2010), originally developed by the IXA group at the University of the Basque Country. Within Wordnet we search the synsets for the incoming word, and the words that correspond to those synsets are returned.

3.4 Rhyme search

The classical and most well-known problem in *bertsolaritza* concern the rhyming patterns. As mentioned, various lines within a verse are required to rhyme, according to certain predefined schemata. To search for words that rhyme with other words in a verse, the BAD tool contains a rhyme search engine. In the interface, this is located in the right part of the BAD tool main view, as seen in figure 2.

The rhyme searcher is built upon finite-state technology, commonly used for developing morphological and phonological analyzers, and calls upon the freely available *foma*-tool, to calculate matching and nonmatching rhyme schemes.

Its grammar is made up of regular expressions that are used to identify phonological patterns in final syllables in the input word. The result of the search is the intersection of these patterns and all the words generated from a morphological description of Basque (Alegria et al., 1996)—that is, a list of all words that match both the required phonological constraints given (rhyming) and a morphological description of Basque.

Based upon figure 2, if we search rhymes for the word *landa* (cottage), the system proposes a

[3] `http://foma.googlecode.com`

Figure 2: The response of the rhyme search engine.

set of words that can be filtered depending on the number of syllables required. Among this list of words, we can find some words that end in *anda*, such as, *Irlanda* (Ireland) or *eztanda* (explosion), but through the application of phonological equivalency rules we also find terms like *ganga* (vault).

3.5 Singing synthesis

Another characteristic, as mentioned, is that, in the end, the verses are intended to be sung instead of only being textually represented. Based on other ongoing work in singing synthesis, we have designed a system for singing the verses entered into the system in Basque.

This is based on the "singing mode" of the Festival text-to-speech system (Taylor et al., 1998). The advantage of using this is that Festival is open-source and has given us ample opportunities to modify its behavior. However, as Festival does not currently support Basque directly, we have relied on the Spanish support of the Festival system.[4]

Based on current work by the Aholab research team in Bilbao—a lab that works on Basque speech synthesis and recognition—we have implemented a singing module for BAD, based on the text-to-speech HTS engine (Erro et al., 2010). Our application is able to sing the composed verses entered into the system in Basque, with a choice of various standard melodies for *bertsolaritza*.[5]

4 Discussion and future work

Now that the BAD tool has been developed, our intention is to evaluate it. To make a qualitative evaluation we have gotten in touch with some verse-making schools (*bertso-eskola*), so that they can test the system and send us their feedback using a form. Once the evaluation is made, we will improve it according to the feedback and the system will be made public.

Our ultimate goal is to develop a system able to create verses automatically. To achieve this long-term goal, there is plenty of work to do and basic research to be done. We have in our hands a good corpus of 3,500 Basque verse transcriptions, so we intend to study these verses from a morphological, syntactical, semantical and pragmatic point of view.

In the short term, we also plan to expand the synonym search to be able to provide searches for semantically related words and subjects (and not just synonyms), like hypernyms or hyponyms. The Basque WordNet provides a good opportunity for this, as one is easily able to traverse the WordNet to encounter words with varying degrees of semantic similarity.

Another feature that we want to develop is a system that receives as input a verse together with a MIDI file, and where the system automatically sings the verse to the music provided.

Finally, in order for the system to be able to provide better proposals for the verse artist—including perhaps humorous and creative proposals—we intend to work with approaches to computational creativity. We are considering different approaches to this topic, such as in the work on *Hahacronym* (Stock et al., 2005) or the *Standup* riddle builder (Ritchie et al., 2001).

[4]While morphologically and syntactically, Spanish and Basque have no relationship whatsoever, phonetically the languages are quite close, with only a few phonemes, syllabification rules, and stress rules being different enough to disturb the system's behavior.

[5]However, this functionality is not available on the web interface as of yet.

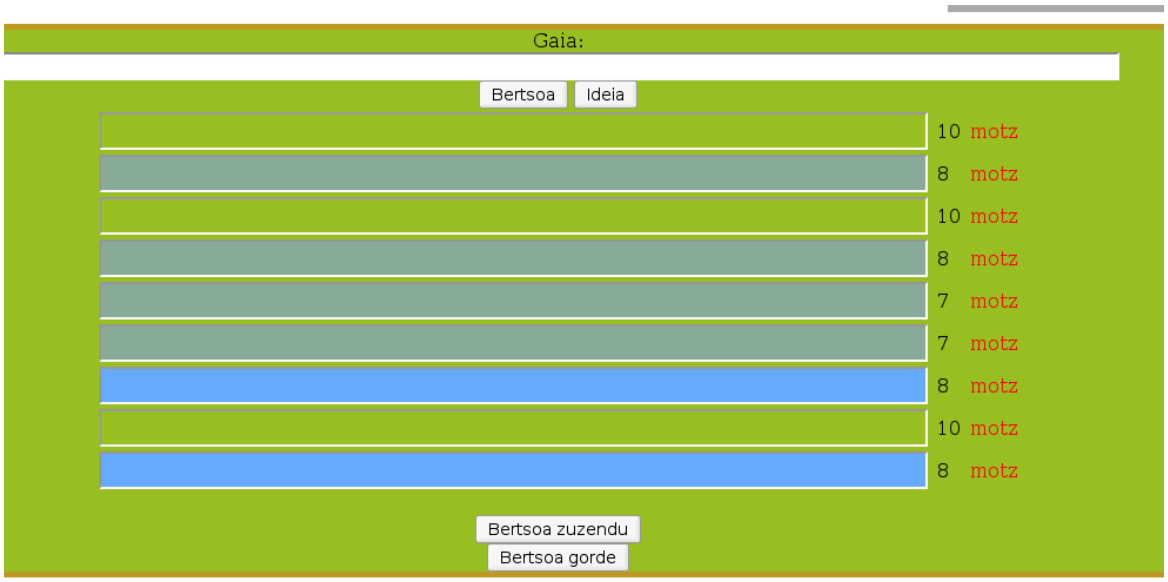


Figure 3: The BAD application before entering a verse, showing two possible rhyme patterns.

Acknowledgments

This research has been partially funded by the Basque Government (Research Groups, IT344-10).

References

Iñaki Alegria, Xabier Artola, Kepa Sarasola and Miriam Urkia, “Automatic morphological analysis of Basque”, Literary and Linguistic Computing, ALLC, 1996

Xabier Amuriza, “Hiztegi errimatua”, Alfabetatze Euskalduntze Koordinakundea, 1981

Bertol Arrieta, Iñaki Alegria, Xabier Arregi, “An assistant tool for Verse-Making in Basque based on Two-Level Morphology”, 2001

Daniel Erro, Iñaki Sainz, Ibon Saratxaga, Eva Navas, Inma Hernáez, “MFCC+F0 Extraction and Waveform Reconstruction using HNM: Preliminary Results in an HMM-based Synthesizer”, 2010

Joxerra Garzia, Jon Sarasua, and Andoni Egaña, “The art of bertsolaritza: improvised Basque verse singing”, Bertsolari liburuak, 2001

John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, “Introducción a la teoría de Autómatas, Lenguajes y Computación”, Pearson educación, 2002

Mans Hulden, “Foma: a finite-state compiler and library”, Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session, p. 29–32, 2009

Mans Hulden, “Finite-state syllabification”, Finite-State Methods and Natural Language Processing, p. 86 – 96, Springer, 2006

Kimmo Koskenniemi. “Two-level morphology: A general computational model for word-form recognition and production”. Publication 11, University of Helsinki, Department of General Linguistics, Helsinki, 1983.

Eli Pociello, Eneko Agirre, Izaskun Aldezabal, “Methodology and construction of the Basque WordNet”, 2010

Graeme Ritchie, “Current directions in computational humour”, Artificial Intelligence Review, Volume 16, Number 2, p. 119 – 135, Springer, 2001

Graeme Ritchie, Ruli Manurung, Helen Pain, Annalu Waller, Dave O’Mara, “The STANDUP interactive riddle builder”, Volume 2, Number 2, p. 67 – 69, IEEE Intelligent Systems, 2006

Oliviero Stock and Carlo Strapparava, “Hahacronym: A computational humor system”, Proceedings of the ACL 2005 on Interactive poster and demonstration sessions, p. 113 – 116, Association for Computational Linguistics, 2005

Paul Taylor, Alan W. Black and Richard Caley, “The architecture of the Festival speech synthesis system”, International Speech Communication Association, 1998

Finite-state technology in a verse-making tool

Manex Agirrezabal, Iñaki Alegria, Bertol Arrieta
University of the Basque Country (UPV/EHU)
maguirrezaba008@ikasle.ehu.es, i.alegria@ehu.es, bertol@ehu.es
Mans Hulden
Ikerbasque (Basque Science Foundation)
mhulden@email.arizona.edu

Abstract

This paper presents a set of tools designed to assist traditional Basque verse writers during the composition process. In this article we are going to focus on the parts that have been created using finite-state technology: this includes tools such as syllable counters, rhyme checkers and a rhyme search utility.

1 The BAD tool and the Basque singing tradition

The BAD tool is an assistant tool for verse-makers in the Basque *bertsolari* tradition. This is a form of improvised verse composition and singing where participants are asked to produce impromptu compositions around themes which are given to them following one of many alternative verse formats. The variety of verse schemata that exist all impose fairly strict structural requirements on the composer. Verses in the *bertsolari* tradition must consist of a specified number of lines, each with a fixed number of syllables. Also, strict rhyme patterns must be followed. The structural requirements are considered the most difficult element in the *bertsolaritza*—however, well-trained *bertsolaris* can usually produce verses that fulfill the structural prerequisites in a very limited time.

The BAD tool presented here is mainly directed at those with less experience in the tradition such as students. One particular target group are the *bertso-eskola*-s (verse-making schools) that have been growing in popularity—these are schools found throughout the Basque Country that train young people in the art of *bertsolaritza*.

The primary functionality of the tool is illustrated in figure 1 which shows the main view of the utility. The user is offered a form in which a verse can be written, after which the system checks the technical correctness of the poem. To perform this task, several finite state transducer-based modules, are used, some of them involving the metrics (syllable counter) of the verse, and others the rhyme (rhyme searcher and checker). The tool has support for 150 well known verse meters.

In the following sections, we will outline the technology used in each of the parts in the system.

2 Related work

Much of the existing technology for Basque morphology and phonology uses finite-state technology, including earlier work on rhyme patterns (Arrieta et al., 2001). In our work, we have used the Basque morphological description (Alegria et al., 1996) in the rhyme search module. Arrieta et al. (2001) develop a system where, among other things, users can search for words that rhyme with an introduced pattern. It is implemented in the formalism of two-level morphology (Koskenniemi, 1983) and compiled into finite-state transducers.

We have used the open-source *foma* finite-state compiler to develop all the finite-state based parts of our tool.[1]. After compiling the transducers, we use them in our own application through the C/C++ API provided with *foma*.

3 Syllable counter

As mentioned, each line in a verse must contain a specified number of syllables. The syllable counter module that checks whether this is the case consists of a submodule that performs the syllabification itself as well as a module that yields variants produced by optional apocope and syncope effects. For the syllabification itself, we use the approach described in Hulden (2006), with some modifications to capture Basque phonology.

[1] In our examples, FST expressions are written using *foma* syntax. For details, visit http://foma.googlecode.com

Figure 1: A verse written in the BAD web application.

3.1 Syllabification

Basque syllables can be modeled by assuming a maximum onset principle together with a sonority hierarchy where obstruents are the least sonorous element, followed in sonority by the liquids, the nasals and the glides. The syllable nuclei are always a single vowel (a,e,i,o,u) or a combination of a low vowel (a,e) and a high vowel (i,o,u) or a high vowel and another high vowel.

The syllabifier relies on a chain of composed replacement rules (Beesley and Karttunen, 2003) compiled into finite-state transducers. These definitions are shown in figure 2. The overall strategy is to first mark off the nuclei in a word by the rule `MarkNuclei` which takes advantage of a left-to-right longest replacement rule. This is to ensure that diphthongs do not get split into separate syllables by the subsequent syllabification process. Following this, syllables are marked off by the `markSyll`-rule, which inserts periods after legitimate syllables. This rule takes advantage of the shortest-leftmost replacement strategy—in effect minimizing the coda and maximizing the size of the onset of a syllable to the extent permitted by the allowed onsets and codas, defined in `Onset` and `Coda`, respectively.

To illustrate this process, supposing that we are syllabifying the Basque word **intransitiboa**. The first step in the syllabification process is to mark the nuclei in the word, resulting in {**i**}**ntr**{**a**}**ns**{**i**}**t**{**i**}**b**{**o**}{**a**}. In the more complex syllabification step, the `markSyll` rule assures that the juncture **ntr** gets divided as **n.tr** because **nt.r** would produce a non-maximal onset, and **i.ntr** would in turn produce an illegal onset in

```
define Obs        [f|h|j|k|p|s|t|t s|t z|t x|x|
                   z|b|d|g|v|d d|t t];
define LiqNasGli  [l|r|r r|y|n|m];
define LowV       [a|e|o];
define HighV      [i|u];
define V          LowV | HighV;
define Nucleus    [V | LowV HighV |
                   [HighV HighV - [i i] - [u u]]];
define Onset      (Obs) (LiqNasGli);
define Coda       C^<4;

define MarkNuclei Nucleus @-> %{ ... %};
define Syll        Onset %{ Nucleus %} Coda;
define markSyll    Syll @> ... "." || _ Syll ;
define cleanUp     %{|%} -> 0;

regex MarkNuclei .o. markSyll .o. cleanUp;
```

Figure 2: Syllable definition

the second syllable. The final syllabification, after markup removal by the `Cleanup` rule, is then **in.tran.si.ti.bo.a**. This process is illustrated in figure 3

In *bertsolaritza*, Basque verse-makers follow this type of syllable counting in the majority if cases; however, there is some flexibility as regards the syllabification process. For example, suppose that the phrase **ta lehenengo urtian** needs to fit a line which must contain six syllables. If we count the syllables using the algorithm shown above, we receive a count of eight (**ta le.hen.en.go ur.ti.an**). However, in the word **lehenengo** we can identify the syncope pattern **vowel-h-vowel**, with the two vowels being identical. In such cases, we may simply replace the entire sequence by a single vowel (ehe $\rightarrow$ e). This is phonetically equivalent to shortening the *ehe*-sequence (for those dialects where the orthographical **h** is silent). With this modification, we can fit

the line in a 7 syllable structure. We can, however, further reduce the line to 6 syllables by a second type of process that merges the last syllable of one word with the first of the next one and then resyllabifying. Hence, **ta lehenengo urtian**, using the modifications explained above, could be reduced to **ta.le.nen.gour.ti.an**, which would fit the 6 syllable structure. This production of syllabification variants is shown in figure 4.

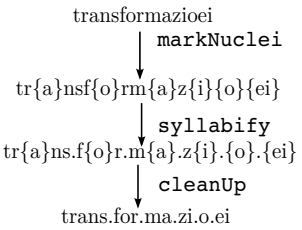


Figure 3: Normal syllabification.

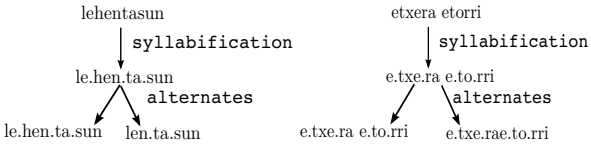


Figure 4: Flexible syllabification.

4 Finite-state technology for rhymes

4.1 Basque rhyme patterns and rules

Similar to the flexibility in syllabification, Basque rhyme schemes also allows for a certain amount of leeway that *bertsolaris* can take advantage of. The widely consulted rhyming dictionary *Hiztegi Errimatua* (Amuriza, 1981) contains documented a number of phonological alternations that are acceptable as off-rhymes: for example the stops **p**, **t**, and **k** are often interchangeable, as are some other phonological groups. Figure 5 illustrates the definitions for interchangeable phonemes when rhyming. The interchangeability is done as a prelude to rhyme checking, whereby phonemes in certain groups, such as **p**, are replaced by an abstract symbol denoting the group (e.g. **PTK**).

4.2 Rhyme checker

The rhyme checker itself in BAD was originally developed as a *php*-script, and then reimplemented as

```
define plosvl      [p | t | k];
define rplosv      [b | d | g | r];
define sib         [s | z | x];
define nas         [n | m];

define plosvlconv  ptk -> PTK;
define rplosvconv  bdgr -> BDGR;
define sibconv     sib -> SZX;
define nasconv     nas -> NM;

define phoRules    plosvlconv .o. rplosvconv .o.
                   sibconv    .o. nasconv ;
```

Figure 5: Conflation of consonant groups before rhyme checking.

a purely finite-state system. In this section we will focus on the finite-state based one.

As the *php* version takes advantage of syllabification, the one developed with transducers does not. Instead, it relies on a series of replacement rules and the special `_eq()` operator available in *foma*. An implementation of this is given in figure 6. As input to the system, the two words to be checked are assumed to be provided one after the other, joined by a hyphen. Then, the system (by rule `rhympat1`) identifies the segments that do not participate in the rhyme and marks them off with "{" and "}" symbols (e.g. **landa-ganga** → <{**l**}**anda**>-<{**g**}**anga**>).

The third rule (`rhympat3`) removes everything that is between "{" and "}", leaving us only with the segments relevant for the rhyming pattern (e.g. <**anda**>-<**anga**>). Subsequent to this rule, we apply the phonological grouping reductions mentioned above in section 4.1, producing, for example (<**aNMBDGRa**>-<**aNMBDGRa**>).

After this reduction, we use the `_eq(X,L,R)`-operator in *foma*, which from a transducer `X`, filters out those words in the output where material between the specified delimiter symbols `L` and `R` are unequal. In our case, we use the < and > symbols as delimiters, yielding a final transducer that does not accept non-rhyming words.

4.3 Rhyme search

The BAD tool also includes a component for searching words that rhyme with a given word. It is developed in *php* and uses a finite-state component likewise developed with *foma*.

Similarly to the techniques previously described, it relies on extracting the segments relevant to the

```
define rhympat1   [0:"{" ?* 0:"}"
  [[[V+ C+]  (V) V] | [(C) V V]] C* ];
# constraining V V C pattern
define rhympat2  ~[?* V "}" V C];
# cleaning non-rhyme part
define rhympat3  "{" ?* "}" -> 0;
define rhympat rhympat1 .o. rhympat2 .o.
  rhympat3;

# rhyming pattern on each word
# and phonological changes
define MarkPattern rhympat .o.
       phoRules  .o. patroiak;
# verifying if elements between < and >
# are equal
define   MarkTwoPatterns
    0:%< MarkPattern 0:%> %-
    0:%< MarkPattern 0:%> ;
define Verify  _eq(MarkTwoPatterns, %<, %>)
regex Verify .o. Clean;
```

Figure 6: Rhyme checking using *foma*.

rhyme, after which phonological rules are applied (as in 4.1) to yield phonetically related forms. For example, introducing the pattern **era**, the system returns four phonetically similar forms **era**, **eda**, **ega**, and **eba**. Then, these responses are fed to a transducer that returns a list of words with the same endings. To this end, we take advantage of a finite-state morphological description of Basque (Alegria et al., 1996).

As this transducer returns a set of words which may be very comprehensive—including words not commonly used, or very long compounds—we then apply a frequency-based filter to reduce the set of possible rhymes. To construct the filter, we used a newspaper corpus, (Egunkaria[2]) and extracted the frequencies of each word form. Using the frequency counts, we defined a transducer that returns a word's frequency, using which we can extract only the n-most frequent candidates for rhymes. The system also offers the possibility to limit the number of syllables that desired rhyming words may contain. The syllable filtering system and the frequency limiting parts have been developed in *php*. Figure 7 shows the principle of the rhyme search's finite-state component.

5 Evaluation

As we had available to us a rhyme checker written in *php* before implementing the finite-state version,

[2] `http://berria.info`

```
regex phoRules .o. phoRules.i .o.
       0:?* ?* .o. dictionary ;
```

Figure 7: Rhyme search using *foma*

it allowed for a comparison of the application speed of each. We ran an experiment introducing 250,000 pairs of words to the two rhyme checkers and measured the time each system needed to reply. The FST-based checker was roughly 25 times faster than the one developed in *php*.

It is also important to mention that these tools are going to be evaluated in an academic environment. As that evaluation has not been done yet, we made another evaluation in our NLP group in order to detect errors in terms of syllabification and rhyme quality. The general feeling of the experiment was that the BAD tool works well, but we had some efficiency problems when many people worked together. To face this problem some tools are being implemented as a server.

6 Discussion & Future work

Once the main tools of the BAD have been developed, we intend to focus on two different lines of development. The first one is to extend to flexibility of rhyme checking. There are as of yet patterns which are acceptable as rhymes to *bertsolaris* that the system does not yet recognize. For example, the words **filma** and **errima** will not be accepted by the current system, as the two rhymes **ilma** and **ima** are deemed to be incompatible. In reality, these two words are acceptable as rhymes by *bertsolaris*, as the **l** is not very phonetically prominent. However, adding flexibility also involves controlling for overgeneration in rhymes. Other reduction patterns not currently covered by the system include phenomena such as synaloepha—omission of vowels at word boundaries when one word ends and the next one begins with a vowel.

Also, we intend to include a catalogue of melodies in the system. These are traditional melodies that usually go along with a specific meter. Some 3,000 melodies are catalogued (Dorronsoro, 1995). We are also using the components described in this article in another project whose aim is to construct a robot capable to find, generate and sing verses automatically.

Acknowledgments

We would like to acknowledge Aitzol Astigarraga for his help in the development of this project. He has been instrumental in our work, and we intend to continue working with him. Also we must mention the Association of Friends of Bertsolaritza, whose verse corpora has been used to test and develop these tools and to develop new ones.

References

Alegria, I., Artola, X., Sarasola, K., and Urkia, M. (1996). Automatic morphological analysis of Basque. *Literary and Linguistic Computing*, 11(4):193–203.

Amuriza, X. (1981). *Hiztegi errimatua [Rhyme Dictionary]*. Alfabetatze Euskalduntze Koordinakundea.

Arrieta, B., Alegria, I., and Arregi, X. (2001). An assistant tool for verse-making in Basque based on two-level morphology. *Literary and linguistic computing*, 16(1):29–43.

Beesley, K. R. and Karttunen, L. (2003). *Finite state morphology*. CSLI.

Dorronsoro, J. (1995). *Bertso doinutegia [Verse melodies repository]*. Euskal Herriko Bertsolari Elkartea.

Hulden, M. (2006). Finite-state syllabification. *Finite-State Methods and Natural Language Processing*, pages 86–96.

Koskenniemi, K. (1983). Two-level morphology: A general computational model for word-form production and generation. *Publications of the Department of General Linguistics, University of Helsinki. Helsinki: University of Helsinki*.

D Bibliografia

Eneko Agirre, Iñaki Alegria, Xabier Arregi, Xabier Artola, Arantza Diaz de de Ilarraza, Montse Maritxalar, Kepa Sarasola, eta Miren Urkia. XUXEN: A spelling checker/corrector for Basque based on two-level morphology. In *Proceedings of the third conference on Applied natural language processing*, pages 119–125. Association for Computational Linguistics, 1992.

Manex Agirrezabal. Bertsotarako arbel digitala [digital board for making verses]. 2011.

Manex Agirrezabal, Inaki Alegria, Bertol Arrieta, eta M. Hulden. BAD: An assistant tool for making verses in Basque. *EACL 2012*, page 13, 2012a.

Manex Agirrezabal, Iñaki Alegria, Bertol Arrieta, eta Mans Hulden. Finite-state technology in a verse-making tool. In *Proceeding of the 2012 conference on Finite-State Methods and Natural Language Processing*, 2012b.

Itziar Aldabe, Maddalen Lopez de Lacalle, Montse Maritxalar, Edurne Martinez, eta Larraitz Uria. Arikiturri: An automatic question generator based on corpora and nlp techniques. In *Intelligent Tutoring Systems*, pages 584–594. Springer, 2006.

Iñaki Alegria, Xabier Artola, Kepa Sarasola, eta Miriam Urkia. Automatic morphological analysis of Basque. *Literary and Linguistic Computing*, 11(4):193–203, 1996.

Iñaki Alegria, Izaskun Etxeberria, Mans Hulden, eta Montse Maritxalar. Porting Basque morphological grammars to foma, an open-source tool. *Finite-State Methods and Natural Language Processing*, pages 105–113, 2010.

Xabier Amuriza. *Bertsolaritza, 1: hitzaren kirol nazionala*. 1981a.

Xabier Amuriza. *Hiztegi errimatua [Rhyme Dictionary]*. Alfabetatze Euskalduntze Koordinakundea, 1981b.

Bertol Arrieta, Inaki Alegria, eta Xabier Arregi. An assistant tool for verse-making in Basque based on two-level morphology. *Literary and linguistic computing*, 16(1):29–43, 2001.

Kenneth R. Beesley eta Lauri Karttunen. Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford*, 2003.

Kim Binsted. Machine humour: An implemented model of puns. 1996.

Dionisio Cañas eta Carlos González Tardón. *¿ Puede un computador escribir un poema de amor?: Tecnorromanticismo y poesía electrónica*. Devenir, 2010.

Joanito Dorronsoro. *Bertso doinutegia [Verse melodies repository]*. Euskal Herriko Bertsolari Elkartea, 1995.

HAP masterra

Bertsozale elkarteko ikerkuntza taldea. *Bertso-estrofak izendatzeko irizpideak*. Euskal Herriko Bertsozale Elkartea, 2009.

Daniel Erro, Iñaki Sainz, Ibon Saratxaga, Eva Navas, eta Inma Hernáez. Mfcc+ f0 extraction and waveform reconstruction using hnm: preliminary results in an hmm-based synthesizer. *Proc. FALA*, pages 29–32, 2010.

Joxerra Gartzia, Andoni Egaña, eta Jon Sarasua. Bat-bateko bertsolaritza: gakoak eta azterbideak. *Euskal Herriko Bertsozale Elkartea*, 2001.

Pablo Gervás. Wasp: Evaluation of different strategies for the automatic generation of spanish verse. In *Proceedings of the AISB-00 Symposium on Creative & Cultural Aspects of AI*, pages 93–100, 2000.

Pablo Gervás. Automatic generation of poetry using a cbr approach. *CAEPIA-TTIA 01 Actas Volumen I*, 2001.

Inma Hernaez, Eva Navas, Juan Luis Murugarren, eta Borja Etxebarria. Description of the ahotts system for the basque language. In *4th ISCA Tutorial and Research Workshop (ITRW) on Speech Synthesis*, 2001.

Mans Hulden. Finite-state syllabification. *Finite-State Methods and Natural Language Processing*, pages 86–96, 2006.

Mans Hulden. Foma: a finite-state compiler and library. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Demonstrations Session*, pages 29–32. Association for Computational Linguistics, 2009.

Ekaitz Jauregi, Jose Maria Martinez-Otzeta, Basilio Sierra, eta Elena Lazkano. Tartalo: the door knocker robot. In *Robotics and Biomimetics, 2007. ROBIO 2007. IEEE International Conference on*, pages 1114–1120. IEEE, 2007.

Elena Lazkano. *Pautas para el desarrollo incremental de una arquitectura de control basada en el comportamiento para la navegación de robots en entornos semi-estructurados*. UPV, Argitalpen Zerbitzua= EHU, Servicio Editorial, 2005.

Robert P. Levy. A computational model of poetic creativity with neural network as measure of adaptive fitness. In *Proccedings of the ICCBR-01 Workshop on Creative Systems*. Citeseer, 2001.

Ruli Manurung. *An evolutionary algorithm approach to poetry generation*. PhD thesis, School of informatics, University of Edinburgh, 2003.

Josef Novak, Nobuaki Minematsu, eta Keikichi Hirose. Wfst-based grapheme-to-phoneme conversion: Open source tools for alignment, model-building and decoding wfst-based Grapheme-to-Phoneme conversion: Open source tools for alignment, model-building and decoding. *Finite-State Methods and Natural Language Processing*, 2012.

HAP masterra

Arantxa Otegi. *Hedapena informazioaren berreskurapenean: hitzen adiera-desanbiguazioaren eta antzekotasun semantikoaren ekarpenak*. PhD thesis, Lengoaia eta Sistema Informatikoak Saila, EHU/UPV. Informatika Fakultatea. 2012/03/16, 2012.

Eli Pociello. Euskararen ezagutza-base lexikala: Euskal wordnet. *Doktoretza-tesia, Euskal Filologia Saila (UPV/EHU). Leioa*, 2008.

Oliviero Stock eta Carlo Strapparava. Hahacronym: A computational humor system. In *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*, pages 113–116. Association for Computational Linguistics, 2005.

Annalu Waller, Rolf Black, David O'Mara, Helen Pain, Graeme Ritchie, eta Ruli Manurung. Evaluating the STANDUP pun generating software with children with cerebral palsy. *ACM Transactions on Accessible Computing (TACCESS)*, 1(3):16, 2009.