eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Informatika Ingeniaritzako Gradua

Konputazioa

Gradu Amaierako Lana

Testu bidezko irudien deskribapenak sortzen

Egilea

Ander Merketegi Badiola

2020

eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Informatika Ingeniaritzako Gradua

Konputazioa

Gradu Amaierako Lana

Testu bidezko irudien deskribapenak sortzen

Egilea

Ander Merketegi Badiola

Zuzendariak

Gorka Azkune

Oier Lopez de Lacalle

Laburpena

Proiektu honetan testu bidezko irudien deskribapenak sortzen dira ikasketa sakoneko zenbait teknika erabiliz. Ikasketa sakonean oinarritutako metodoak ondo ulertzeko, beharrezko oinarri teorikoaren azterketa sakona egin da. Behin oinarri teoriko hau finkatuta, helburua irudiak errepresentatzeko bi eredu alderatzea izan da. Horretarako, esperimentazioa gauzatu da, eta hainbat hiperparametroren esplorazio bat egin bi sistemen arteko aldea ikusi eta biak konparatu ahal izateko. Amaitzeko, lorturiko emaitzak ikusi eta hauen desberdintasunen inguruko ondorioak atera dira. Ondorio bezala, ez da alde nabarmenik ikusi bi ereduen artean, baliteke erabilitako datu-multzo txikia dela eta. Azkenik, etorkizun batean interesgarriak izan litezkeen zenbait ataza ere zehaztu dira, proiektua aberasteko asmotan.

Gaien aurkibidea

Irudien aurkibidea

Taulen aurkibidea

1. KAPITULUA

Sarrera

1.1 Motibazioa

Egunero telebista, artikulu edo internet bezalako iturrietan ehunka eta ehunka irudirekin egiten dugu topo. Irudi hauek, oro har, ikusleak berak interpretatu behar izaten ditu, izan ere, ez dute deskribapen zehatzik ekartzen. Hala ere, gizakia gai da hauek arazo handirik gabe ulertzeko eta interpretatzeko, jada prozesu inkontziente bat bilakatu duelarik.

Jar gaitezen orain, ordea, itsu baten lekuan. Argazki hauek ezin ikusi ahal izatetik haratago, egunerokotasunean ere ezer ezin ikus dezakeen pertsona baten azalean. Imajinatu, denbora errealean begiratzen duten guztia automatikoki deskribatzen duen sistema bat, beraien eran hauek ere mundua ikus dezaten. Oinarrian, bideo baten deskribapen automatiko bat egitearen antzekoa izango zen, ezta? Eta hare oinarrizkoago, argazkien deskribapen automatikoa.

Gaur egun, garatu dira irudien deskribapenak modu automatikoan sortzeko adimen artifizialeko ereduak. Egitura hauek sare neuronalak dituzte oinarrian, eta hain zuzen modu horretako arkitektura bat aztertuko da memoria honetan.

Gizakiak hau automatikoki egiten badu ere, makinek irudiaren aldaera eta forma desberdinak interpretatu behar dituzte. *Facebook* eta *Twitter* bezalako sare sozialek, esaterako, sarri erabiltzen dituzte testu bidezko irudien deskribapenak lortzeko teknikak. Horrela, argazkiak deskriba ditzakete non gauden, zer jantzi dugun eta zer egiten ari garen esanez. Bestalde, aplikazio ugari ditu biomedikuntza, hezkuntza eta merkataritza bezalako

sektoreetan.

Arlo zientifikoari dagokionez, adimen artifizialean ere punta-puntako ikerkuntza atala da *image captioning* edo testu bidezko irudien deskribapenaren sorkuntza automatikoa. Izan ere, arlo honek hizkuntza prozesamendua eta konputagailu bidezko ikusmena biak uztartzen ditu. Alde batetik, irudien gaineko objektu detekzio bat egitea beharrezkoa da, objektu hauen propopietate eta hauen arteko erlazioak erauziz. Bestetik, ezaugarri hauekin zentzua duten eta gramatikalki zuzenak diren perpausak sortu behar dira, irudiaren nondik norakoak islatuz.

Proiektu honetan, irudietatik testu bidezko deskribapenak sortzeko arkitektura baten berrinplementazioa egiten da, ikasketa sakoneko teknikak erabiliz. Horretarako, ereduaren funtzionamendu nahiz intuizioa ulertzeko beharrezkoak diren oinarri teorikoen azterketa sakona egin da, modelo eta teknika berriak ikasiz eta nireganatuz.

1.2 Irudien testu deskribapenaren sorkuntza

Irudi berarentzat hainbat deskribapen desberdin egon daitezke. Honek ez du esan nahi bata beste baino hobea denik. Hauetaz gain, ziurrenik irudia deskriba dezaketen esaldi gehiago ere egon litezke. Azpiko 1.1 irudian esaterako, *"txakur bat korrika dabil belarretan"*, *txakur marroi bat saltoka zelaian"* edo *"kolore marroiko txakur bat jolasean dabil zelai batean"* bezalako deskribapenak esanguratsuak izan daitezke irudi berarentzat. Guretzat hain sinplea den ataza hau ordea, ez da horren tribiala konputagailu batentzat.

Sare neuronal eta ikasketa sakonaren azken urteotako aurrerapena baino lehen, pentsaezina zen konputazio ikusmeneko ikerlarientzat ordenagailua gai izango zenik datu-multzo egoki batekin arazo honi aurre egiteko. Gaur egun hau posible da neurona sare arkitektura konplexuen bidez. Hala ere, paradigma honen benetako garrantzia ikusteko bizitza errealeko zenbait aplikazio aipatuko ditut.

- Segurtasun kamerak: Segurtasun kamerak (*CCTV cameras*) nonahi aurki ditzakegu gaur egun. Ikusten duten guztiaz gain ikusten duten hori deskribatzeko ahalmena izanez gero, edozein arazo edo ezustekoren aurrean alarma egoera piztu ahal izango zen momentuan bertan. Honek krimen eta istripu ugari sahiestea ekarri ahalko luke.
- Ibilgailu autonomoak: Ibilgailu autonomoen egungo erronkarik handienetako bat da. Ibilgailuaren inguruko eszena modu seguru eta egonkor batean deskribatzea lortuz gero, bultzada sendoa eman diezaioke.

1.1 Irudia: Irudi berak hainbat deskribapen izan ditzake

Iturria: https://tractive.com/static/images/product-images/trdog1/dog-tracker-honey.jpg

- Itsuei laguntza: Aurrez aipatu bezala, itsuei libreago ibiltzeko eta bidaiatzeko aukera eman diezaieken sistema bat sor liteke. Horretarako, eszenaren testu errepresentazioa lortu beharko litzateke, ondoren audio bihurtuz, adibidez.

Hauetaz gain beste hainbat eta hainbat aplikazio ditu arlo ugaritan, eta arazo ugariri aurre egiten lagun dezake. Hala eta guztiz ere, ez da ataza erraza ordenagailu batek modu automatikoan irudi baten deskribapen natural bat sortzea, eta erronka puntu horrek ikerlari ugariren interesa piztu du. Izan ere, konputagailu bidezko ikusmena eta hizkuntza prozesamendua biak lotuz nolabait, ez da nahikoa irudi baten eduki semantikoaren ulermen maila altua izatea, baizik eta eduki hori modu gizatiar eta natural batean adieraztea ere ezinbestekoa da.

Beraz, garrantzitsua da *image captioning* sistema batentzat objektu eta eszenaren propietate desberdinak identifikatzeaz gain, informazio hau modu natural batean adieraztea. Denboran zehar modelo eta arkitektura desberdinak proposatu dira, eta hobekuntza nabarmenak izan ditu oro har. Esaterako, [Kojima et al., 2002] artikuluan, bulego bateko langileen portaera deskribatzeko sistema bat aurkeztu zen, ingurumen finko batean. Horretarako, gertakari edo kasu egiturak, ekintzen hierarkia kontzeptualak eta aditzen arteko patroiak erabili ziren. Hala ere, metodo hauek gaur egun aurki ditzakegun aplikazioetatik urrun zeuden oraindik.

Urte batzuk beranduago jada bizitza errealeko argazki naturalagoetan oinarritzen ziren sistemak lantzen hasi ziren ([Farhadi et al., 2010] adibidez), baina ez da orain dela gutxi

arte izan sare neuronal sakonen bidezko *image captioning* sistemek indarra hartu dutela, adimen artifizial eta sare neuronalen hobekuntzari esker. Hortik aurrera, geroz eta aplikazio gehiago sortu dira, eta etengabe aurrera doan arloa da, zailtasun ugariko gaia izan arren.

1.3 Proiektuaren helburuak

Proiektu honen helburu nagusia testu bidezko irudien deskribapenak sortzeko bi arkitektura desberdin sakonki aztertu, probatu eta hauen arteko konparaketa bat egitea izan da. Horrela, bien egitura, desberdintasun eta emaitzak alderatuz, hauen arteko diferentzia ikusi ahal izango da. Helburu hau aurrera eramatearren, ordea, hainbat azpi-helburu edo eginkizun bete behar izan dira, eta hauek ere proiektuaren eta ikasketaren zati handi bat izan dira.

- Literatura aztertu: Erabilitako arkitekturak konplexuak izan direnez, hauek ondo ulertzeko aldez aurretik oinarrizko modelo eta egiturak errepasatzea beharrezkoa da. Gainera, adimen artifizial eta *image captioning* arloan berria izanik, ataza hau ondo burutzea ezinbestekoa izango da, eta arrazoi hau dela eta, ikasketa eta errepasoa egitura konplexu hauen pieza edo atal funtsezkoenetik hasi da.
- Arkitekturak definitu: Proiektua burutzeko arkitekturak definitu eta aukeratu behar izan dira. Horretarako, aurrez, artikulu eta literatura desberdinak aztertu dira, eta azkenik, alderatzeko modukoak diren bi sistemak zehaztu. Ereduen arteko desberdintasun nagusia kodetzailean dago: lehen sisteman, irudia bere osotasunean hartzen da ezaugarriak hartzeko; bigarrenean, aldiz, irudian zeharreko elementuak identifikatzen saiatzen da irudia kodetzeko.
- Irudi-kodetzailearen informazioaren errepresentazio azterketa: Aurreko helburuarekin lotuta, bi irudi kodetzaileren informazio-errepresentazio desberdin aztertuko dira, eta bi kodeketak alderatuko dira baldintza esperimental berdinetan, irudiak errepresentatzeko erabilitako egituren arteko desberdintasunak ikusteko.
- Datu-multzoa identifikatu: Tesu bidezko irudien dekribapenak sortzeko, beharrezkoa eta ezinbestekoa da datu-multzo esanguratsua izatea, ataza konkretu honetarako baliozkoa dena. Datu-base egoki bat aukeratzea beharrezkoa da ondoren sarearen entrenamentu eta emaitzak esanguratsuak izateko. Gainera, erabiliko den datu-multzoa publikoa izango da.

- Hiperparametroen azterketa: ereduak entrenatzerako garaian eskura ditugun hiperparametroen azterketa egingo da. Horrela, sistema hobetzen saiatzeaz gain, hauek entrenamendu garaian izan dezaketen eragina ikusi ahal izango da konbinaketa desberdinak probatuz.

1.4 Dokumentuaren egitura

Memoria honetan proiektuaren nondik norakoak azaltzen dira eta kapitulu desberdinetan antolatuta dago. 2 kapituluan, ikasketa sakonaren bidez irudien testu-deskribapenen sorkuntza automatikorako erabiltzen diren teknikak, arkitekturak eta hauen xehetasunak behar bezala ulertzeko oinarri teorikoak azalduko dira. 3 atalean, erabili diren bi ereduen diseinuak sakonki azalduko dira, nolabait biak alderatuz, eta erabilitako hiztegi zein entrenamendu teknikak zehaztuko dira. Ondoren, 4 kapituluan, bi sistemen ebaluazioa aztertuko da, bai eta lorturiko emaitza desberdinak alderatu eta hauen inguruan eztabaidatu ere. Amaitzeko, 5 kapituluan proiektuak eragindako ondorioak nahiz etorkizunerako lana zehaztuko dira.

2. KAPITULUA

Ikasketa sakonaren bidezko irudien testu deskribapena

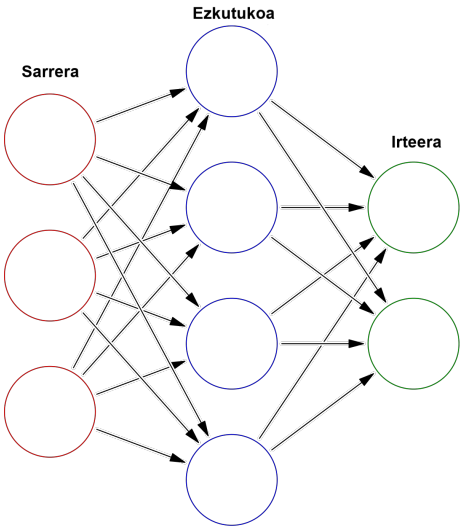
2.1 Sare neuronalak

Sare neuronal aritifizialak burmuin biologikoen portaera eta funtzionamenduan inspiraturiko eredu konputazionalak dira. Sare hauen oinarrizko unitateak neurona artifizialak dira, neurona biologikoen portaera simulatzen dutenak, eta hauen arteko konexioen bidez sortzen dira neurona sare artifizialak. Hau horrela, grafo zuzendu baten gisan adieraziko dugu sarea, non sarrera bat emanda irteera bat sortzen duen (2.1 irudia).

Neuronak modu hierarkikoan antolatzen dira, hau da, geruza berean dauden neuronek sarrera berdinak jasotzen dituzte. Neuronen arteko loturek neurona baten irteerak hurrengo neruonan duen pisua adierazten dute. 2.2 irudian ikus daitekeen bezala, ezkutuko geruzako $h1$ nodoak sarrerako $i1$, $i2$ balioak jasoko ditu bakoitza bere $w1$ eta $w2$ pisuekin, baita 1 balioa duen $b1$ atalasea ere.

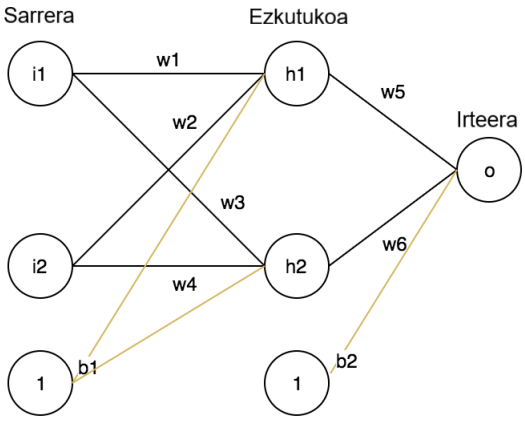
Neurona artifizial bakoitzak sarrera hauek jaso, dagozkien pisuekin biderkatu eta atalasea gehitzen die, azkenik aktibazio funtzio batetik pasatzeko ez-lineartasuna ezarriz. Horrela, lorturiko irteera hau izango da hurrengo geruzako neuronen sarrera. 2.3 irudian, neurona baten egitura ikus daiteke, non $(x_0, x_1, ..., x_n)$ sarrerak izango diren, x_0 atalasea izanik, eta $(w_0, w_1, ..., w_n)$ dagozkien pisuak.

Hau kontuan izanik, neurona batek betetzen duen funtzioa honakoa izango da, ϕ aktibazio



2.1 Irudia: Sare neuronal bateko sarrera geruza, ezkutuko geruza eta irteera geruza. Geruza bakoitzeko neurona guztiak hurrengo geruzako eta aurreko geruzako neurona guztiekin lotuta daude.

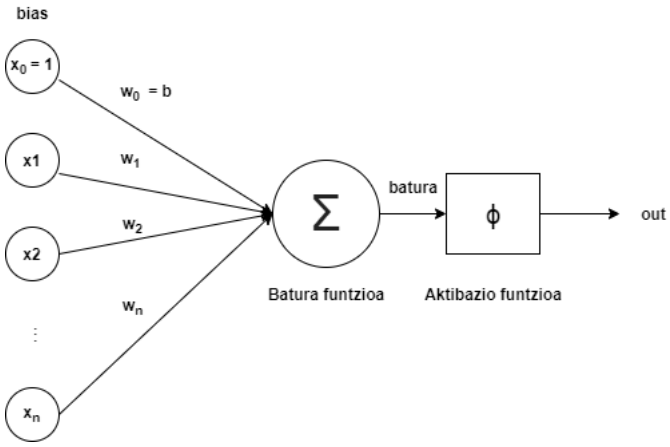
Iturria: https://es.wikipedia.org/wiki/Red_neuronal_artificial#/media/Archivo:Colored_neural_network_es.svg



2.2 Irudia: Sare neuronal baten sarrera geruza, ezkutuko geruza eta irteera geruza. Neuronen arteko loturek w_i pisuak adierazten dituzte, sarearen parametroak.

funtzioa izanik:

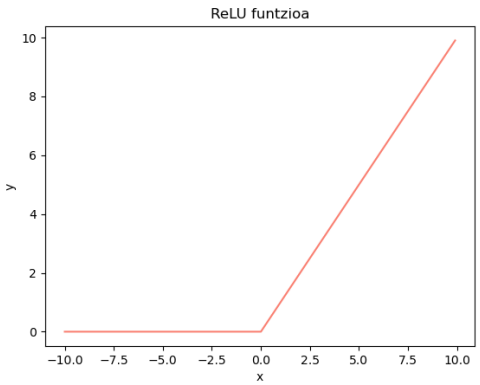
$$out = \phi(\sum_{i=0}^{n} w_i x_i) \tag{2.1}$$



2.3 Irudia: Neurona artifizial baten egitura. $\sum$ batura funtzioak x_i sarrera balio guztiak batuko ditu, hauek dagozkien w_i pisuekin biderkatu ondoren. Azkenik, batura hau ϕ aktibazio funtziotik pasako da irteera balioa eskuratuz.

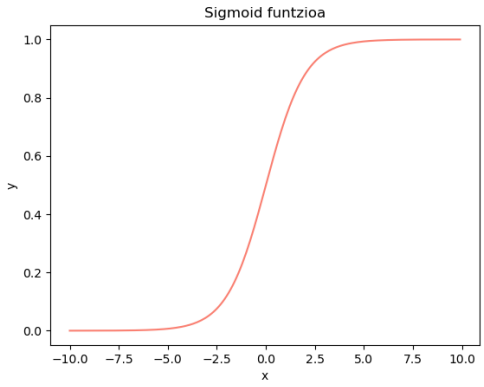
Aktibazio-funtzioak ez-lineartasunaren propietatea sartzen du gure sarean, eta propietate hau ezinbestekoa da sareak ikasteko ahalmena izan dezan. Neurona artifizial baten ϕ aktibazio-funtzio gisa hainbat funtzio desberdin erabil daitezke:

- **ReLU** funtzioa
  - $ReLU : \mathbb{R} \to [0, \infty)$, $\qquad ReLU(x) = \max(0, x)$



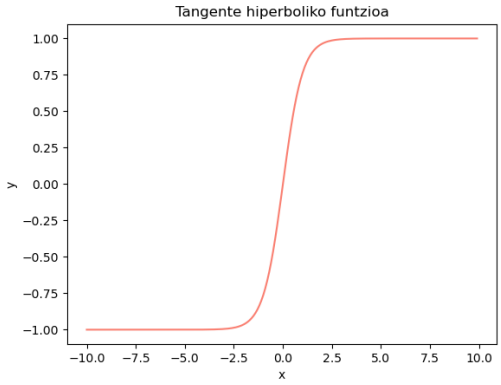
2.4 Irudia: ReLU funtzioa.

- **Sigmoid** funtzioa
  - $\sigma : \mathbb{R} \to (0,1)$, $\sigma(x) = \dfrac{1}{1+e^{-x}}$



2.5 Irudia: Sigmoid funtzioa.

- **Tangente hiperboliko** funtzioa
  - $\tanh : \mathbb{R} \to (-1,1)$; $tanh = \dfrac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$



2.6 Irudia: Tangente hiperboliko funtzioa.

2.1.1 Sare neuronalen ikasketa

Sare neuronalek, irteera balioak sortzeko, sarrera bat jaso eta neuronak lotzen dituzten w_i pisuekin biderkatzen dute hau, ondoren denak batu eta aktibazio funtziotik pasatzeko. Beraz, sarearen irteeera balioek zerikusi zuzena dute neuronak lotzen dituzten pisuekin.

Sare bateko pisu guztien $\theta = \{w_i\}$ multzoari sarearen parametro multzoa deritzo, eta parametro hauen balioaren arabera emaitza aldatzen joango da. Hau horrela, ebatzi nahi den problema edo ataza gauzatzeko parametro hauen ahalik eta balio edo konbinaziorik onenak lortu beharko dira, eta hauek lortzeko prozesuari deitzen zaio sare neuronalaren ikasketa edo entrenamendua.

Sarea entrenatzeko beharrezkoa da datu-multzo bat izatea. Datu hauek sarearentzat hasieran ezezaguna den distribuzio bat jarraituko dute, eta sarea bere parametroak egokitzen joango da, pixkanaka distribuzio hau ikasiz.

Ataza eta ikasketarako eskuragarri dagoen datu-multzoaren arabera, hiru ikasketa paradigma bereiz daitezke:

- **Ikasketa gainbegiratua:** (x, y) gisako sarrera-irteera pareak ditugu datu gisa, eta helburua bi hauen arteko lotura modelatzea da.
- **Ikasketa ez gainbegiratua:** sarrerako datuak soilik daude eta ez dute etiketaturiko irteerarik. Helburua sarrerako datuen gaineko propietate edo erlazioa aurkitzea da.
- **Errefortzu bidezko ikasketa:** agente batek hainbat sarrera datu jasoko ditu bere ingurunetik. Helburua agenteak gauzatu beharreko ekintzak ikastea da, hartutako erabakien arabera ematen den "saria"maximizatuz.

Kostu-funtzioak

Sarearen parametroak hobetu ahal izateko, eta hauek ebatzi nahi den atazarako egokiak diren edo ez jakiteko, parametro multzo hau zenbaterainoko ona den esango digun funtzio bat beharrezkoa da, eta hori kostu-funtzioa da. Beraz, kostu-funtzioa sare neuronala ebaluatuko duen funtzioa izango da, eta honek, lortutako irteeraren eta esperotako irteeraren arteko errorea kalkulatuko du. Hau kontuan izanik, problemaren arabera garrantzitsua izango da kostu-funtzio egokia hautatzea.

Gure problema ebazteko erabili den kostu funtzioa **entropia gurutzatua** izan da.

Entropia gurutzatua sailkapen problematan gehien erabiltzen den kostu-funtzioa da. Klase anitzeko sailkapen problemetan, neurona sarearen irteera klase hauen guztien gaineko probabilitate distribuzio gisa uler daiteke. Aurresandako klaseen eta benetan dagozkien klaseen arteko diferentzia zenbat eta txikiagoa izan, entropia gurutzatuaren balioa ere txi-

kiagoa izango da. H entropia gurutzatua honakoa izango da y_i i.garren sarrerari dagokion irteera izanik eta $\hat{y}_i$ sarearen irteeran lorturikoa:

$$H(y,\hat{y}) = -\sum_i y_i log\hat{y} \tag{2.2}$$

Beraz, sarearen ikasketarako errore hau minimizatzea komenigarria izango da. Entrenamenduan, instantzia guztien entropia gurutzatuaren balioak batu eta balio hauen batezbestekoa erabiltzen da kostu-funtzio gisa, aurresandako probabilitate distribuzioa eta benetako probabilitate banaketa ahalik eta antzekoenak izan daitezen.

$$H(\{y^{(n)}\},\{\hat{y}^{(n)}\}) = \frac{1}{n}\sum_n H(y^{(n)},\hat{y}^{(n)}) \tag{2.3}$$

Optimizazio algoritmoak

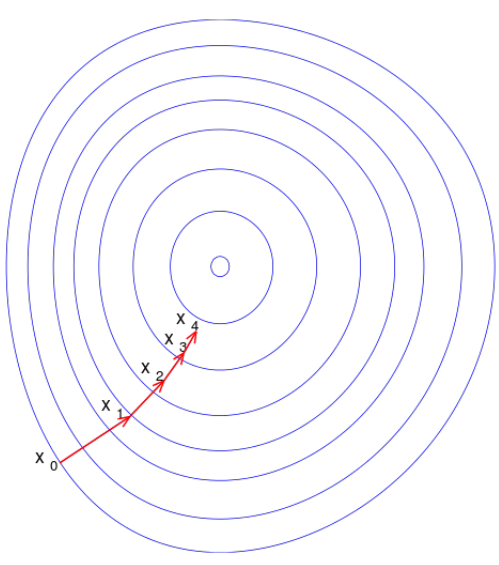
Orokorrean, optimizazio algoritmoak funtzio baten maximo edo minimo globalak aurkitzeko erabiltzen dira. Sare neuronaletan, kostu-funtzioa minimizatzeko erabiltzen dira, irteeran lorturiko emaitzen eta emaitza errealen arteko errorea ahalik eta txikiena izan dadin. Neurona sareak optimizatzeko **gradientearen jeitsiera** erabiltzen da sarritan.

Gradientearen jeitsiera gradientearen kalkuluan oinarritzen da. Gradientea, aldagai anitzeko funztio batean, honen deribatu partzialek osatzen duten matrize bat izango da, eta honek funtzioaren hazkunde handiena adierazten duen norantza zehazten du. Beraz, algoritmoaren eraginez, norabide horren aurkako norantzan mugituko da, ikasketa-tasaren arabera. 2.7 irudian ikus daitekeen gisan, maila-multzoka antolaturiko gainazal batean, pixkanaka minimorantz hurbiltzen joango da, eta x_0-tik x_1-erako saltoaren luzera da, esaterako, ikasketa-tasak zehaztuko duena.

Ezinbestekoa da gradientearen kalkulu hau modu eraginkor batean egitea, izan ere, sare neuronal baten entrenamenduan egin beharreko kalkulu kantitatea oso handia da. Horretarako, [Rumelhart et al., 1986] artikuluan proposaturiko *backpropagation* algoritmo ezaguna erabiltzen da, katearen erregelan eta programazio dinamikoan oinarritua.

Optimizazio algoritmoen oinarria zehazturik, erabiltzen diren algoritmo asko gradiente jeitsieraren aldaera desberdinak dira, estaterako *stochastic gradient descent (SGD)* eta *Adam* optimizazio algoritmoak.

Stochastic Gradient Descent edo gradientearen jeitsiera estokastikoan [Bottou, 2010],



2.7 Irudia: Gradiente jeitsiera, maila-multzo batean adierazia.

Iturria: https://en.wikipedia.org/wiki/Gradient_descent#/media/File:Gradient_descent.svg

entrenamendu-multzoko adibide bakoitzaren gradientea kalkulatu, eta gradiente horrekin sareko parametro guztiak eguneratzen dira; hau da, datu-baseko adibide edo *sample* guztiak hartu, adibide bakoitzeko pisu guztien gradienteak kalkulatu eta hauen batezbestekoa erabili beharrean eguneraketak egiteko, adibide bakar bat hartu, gradientea kalkulatu eta gradiente hori erabiliko da sareko pisu guztiak eguneratzeko, hau datu-multzoko adibide bakoitzarekin egingo delarik.

Beste era batean esanda, adibide bakoitzeko eguneraketa bat gauzatuko dela esan genezake, eguneraketa orokor bakar bat egin beharrean. Modu honetan, datu-multzoa handia denean, ez da hau memorian kargatu beharrik izango.

Proiektuan bi hauen arteko konbinaketa bat erabiltzen da, **Mini-batch Stochastic Gradient Descent** deitua. Kasu honetan, sorta bakoitzerako, sortako lagin bakoitzarekin gradientea kalkulatu eta sorta bakoitzeko gradientearen batezbestekoa kalkulatzen da, balio honekin sareko parametroak eguneratuz.

Adam optimizatzailea ([Kingma and Ba, 2015]) gradientearen jeitsiearaen beste aldaera bat da, gure proiekturako erabili dena zehazki. Gradientearen jeitsiera estokastikoan, ikasketa-tasa bera mantentzen da parametro guztien eguneraketarako, eta ez da aldatzen entrenamendu osoan. Adam optimizatzaileak, ordea, ikasketa-tasa desberdinak konputa-

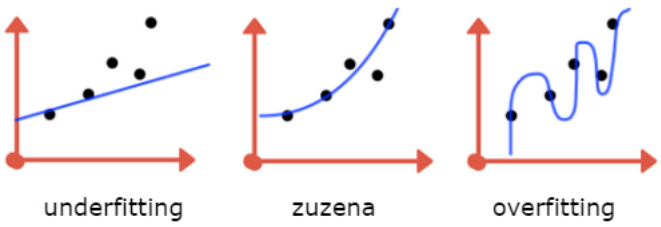
tzen ditu sarearen parametro desberdinetarako, hauek egokitzeko gaitasuna izanik. Hau burutzeko, gradientearen lehen eta bigarren momentuen estimazioak erabiltzen ditu.

Ereduaren orokortzea

Eredua entrenatzearen helburua ahalik eta errore txikiena lortzea da lortutako emaitzen eta erreferentzien artean. Baina gerta liteke, entrenamenduko distribuzioaren gainean burututiko ikasketa ona izan arren, distribuzio bera jarraitzen ez duten datu berriren bat emanez gero, kostu-funtzioa altua izatea. Neurona sareak behar bezala ebaluatzeko *holdout* teknika erabiltzen da, hots, hasierako datu-multzoa hiru zatitan banatzen da: *train*, *dev* eta *test*.

Train zatia da entrenamendua egiteko erabiliko dena. Epoka bakoitzean, hemengo adibide guztiak erabiliko dira, lortutako balioak balio originalekin alderatu eta kostu-funtzioaren kalkulu bidez sareko parametroak eguneratzeko.

Epoka bakoitzaren ostean, pisuak eguneratzean, ebaluazioa gauzatzen da *dev* multzoko adibideekin. Gerta liteke, entrenamendu zatian lorturiko kostu-funtzioaren balioa baxua izatea, baina ez balidazio zatian; hau da, entrenamenduko datuak oso ondo ikasi eta ondoren balidazio adibideen gainean orokortzeko gai ez izatea. Fenomeno honi *overfitting* deitzen zaio. Bestetik, batzuetan entrenamendu errorea altua izaten da eta ez da adibideetara ondo egokitzen; hau da, ez da distribuziora ondo egokitzen. Honi *underfitting* deritzo. Hauek argiako ikus daitezke 2.8 irudian.



2.8 Irudia: *Underfitting:* funtzioak ez du distribuzioa behar bezala ikasi eta ez da orokotzeko gai; Zuzena: funtzioa datuen distribuziora egokitu da; *Overfitting:* Funtzioa entrenamendu datuetara gehiegi egokitu da eta ez da kasu berriak behar bezala orokortzeko gai.

Iturria: https://www.aprendemachinelearning.com/que-es-overfitting-y-underfitting-y-como-solucionarlo/

Azkenik, hainbat epoka gauzatu ondoren eta parametroak ahalik eta ondoen egokituta, ereduaren benetako ebaluazioa *test* zatiaren gainean egingo da.

2.2 Oinarrizko arkitekturak

Ebatzi behar den atazaren arabera, sare neuronalen arkitektura desberdinak definitzen dira, hauetara ahalik eta ondoen egokitzeko. Hauek dira gehien erabiltzen diren arkitekturetako batzuk:

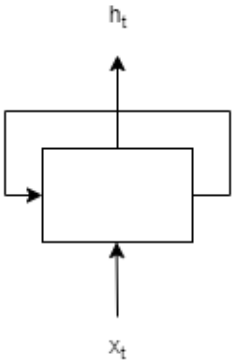
2.2.1 Multilayer perceptron

Arkitekturarik ezagunena eta oinarrizkoena da *multilayer perceptron (MLP)*. 2.1 irudian ikus daitekeen bezala, neuronak geruzetan antolatzen dira, eta neuronen arteko loturek ez dute ziklorik sortzen. Lehen geruza sarrerako geruza da; azkena, irteerako geruza eta tartekoak, berriz, ezkutuko geruzak. Geruza bateko nodo guztiak aurreko geruzako nodo guztiekin lotuta daude.

2.2.2 Sare errekurrenteak

Sare neuronal errekurrenteetan, aurreko urratseko irteera uneko urratseko sarrera bezala erabiltzen da. Sare neuronal tradizionaletan, sarrera eta irteera guztiak elkarrengandik independienteak dira. Esaldi batean hurrengo hitza aurresan nahi bada ordea, esaterako, beharrezkoa da aurreko hitza zein izan den jakitea eta beraz, sareak nolabaiteko memoria izatea. Sare errekurrenteen atzeranzko konexioek posible egiten dute hau, eta honek sekuentziak analizatzeko gaitasuna ematen dienez, maiz erabiliak dira hizkuntzaren prozesamenduan. Azpiko 2.9 irudian sare errekurrentea kutxa bat bezala adieraziko da, non x_t t uneko sarrera emanda h_t irteera emango duen denbora-une horretarako. Irudian ikus daitekeen bezala, begiztaren bidez informazioa pasa ahal izango da urrats batetik bestera.

Sare errekurrente bat azaltzeko ohiko modua sarea hedatuta edo "destolestuta"ikustea da (2.10 Irudia). Uneko $x_t \in \mathbb{R}^n$ sarrera eta aurreko $h_t \in \mathbb{R}^d$ egoera ezkutua dagozkien parametro matrizeekin biderkatu ($W_x \in \mathbb{R}^{d\times n}$ eta $W_h \in \mathbb{R}^{d\times d}$, hurrenez hurren) eta hauei



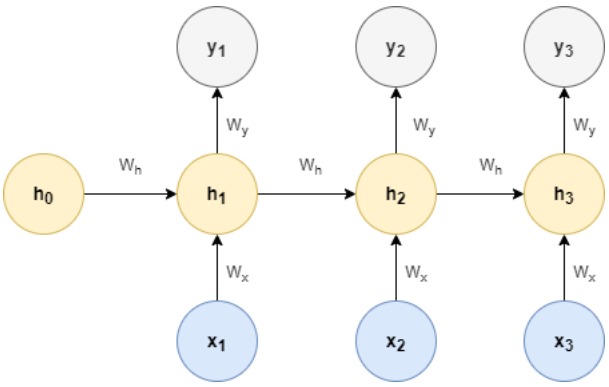
2.9 Irudia: Sare errekurrentea kutxa moduan adierazita, x_t t uneko sarrera eta h_t irteera izanik.

atalasea batu ondoren, σ_h funtzio ez lineal bat aplikatzen zaie, horrela egoera ezkutu berria lortuz:

$$h_t = \sigma_h(W_x \cdot x_t + W_h \cdot h_{t-1} + b) \tag{2.4}$$

Eta irteera lortzeko ezkutuko egoera hau erabiliko da, dagokion W_y matrizearekin biderkatu eta honi ere atalasea batuz, azkenik σ_y aktibazio-funtziotik pasatzeko:

$$y_t = \sigma_y(W_y \cdot h_t + b) \tag{2.5}$$



2.10 Irudia: Sare errekurrentea hedatuta.

Gradienteen eguneraketari dagokionez, *backpropagation through time (BTT)* algoritmoa erabiltzen da. Aurrez aipaturiko [Rumelhart et al., 1986] oinarrizko atzeranzko propagazioaren funtzionamendu berdina du, kasu honetan sare errekurrenteei aplikatuta, [Werbos, 1990]

artikuluan aurkezten den moduan. Laburbilduz, sare errekurrenteetan, hainbat irteera sortzen direnez, irteera guztien erroreen gaineko batura izango da kostu-funtzioa.

Hala ere, badute arazo bat sare errekurrente arruntek. Pausu bakoitzean nolabait geruza bat gehitzen denez, oso sakonak dira eta gradientea oso txikia denean, atzeranzko propagazioan gradiente hau behin eta berriro biderkatzen denez geruza bakoitzean, errorea 0 baliorantz joango da. Honi *vanishing gradients* arazoa deritzo. Gradientearen balioa, oso altua denean, berriz, ekartzen duen efektua erabat aurkakoa da. Honi *exploiding gradients* esaten zaio.

Arazo hauei aurre egiteko *gate units* edo "ateen"egituran oinarrituriko bi arkitektura sortu ziren: *Long short-Term Memory (LSTM)* eta *Gated Recurrent Units (GRU)*.

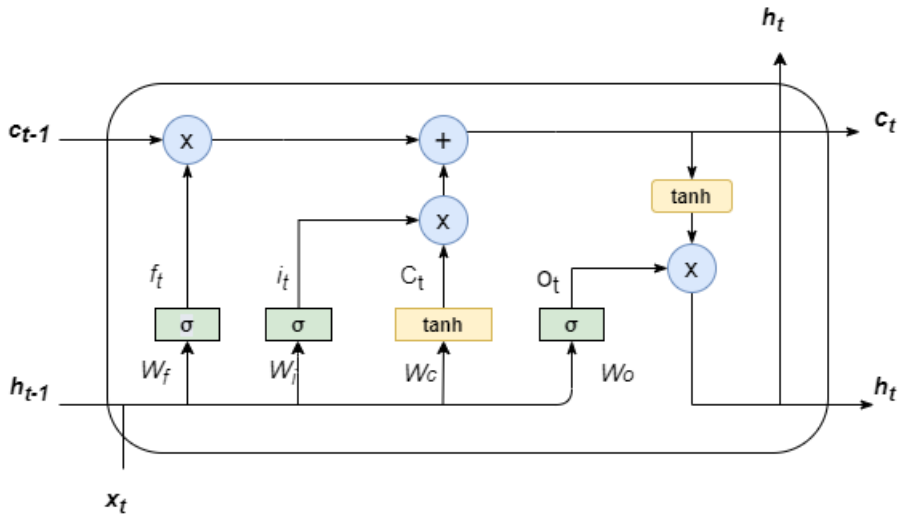
Long Short-Term Memory (LSTM)

Long Short-Term Memory arkitektura [Hochreiter and Schmidhuber, 1997] artikuluan aurkeztu zen 1997 urtean. Honen bidez aipaturiko *vanishing gradients* eta *exploiding gradients* arazoak desagertzen dira. Arkitektura honek informazioa epe luzean zehar gordetzea ahalbidetzen du, eta hori dela eta, testuinguru zabalagoa behar duten atazetan asko erabiltzen da.

Hau ahalbidetzen duen egitura nagusiak zelula-egoera eta *gated units* ateak dira. Sare errekurrente sinple batek tanh geruza bat soilik izan dezakeen bezala, *LSTM*-en egitura konplexuagoa da. Hona hemen 2.11 irudian, LSTM zelula baten egituraren irudian, agertzen diren elementu desberdinak:

- x_t : t uneko sarrera
- h_t : t uneko egoera ezkutua
- c_t : t uneko zelula egoera
- f_t : t uneko ahazteko atea (*forget gate*)
- i_t : t uneko sarrera atea
- σ_t : t uneko irteera atea
- W_f, W_i, W_c, W_σ sarearen parametroak

Zelula egoera informazioa garraiatzen duen "zinta"bat bezala ikus liteke. Honek, hainbat operazio linear soilik jasotzen ditu, eta oso erraza da beraz informazioa bertatik garraiatzea. *LSTM* sareek zelula egoerara pasako den informazioa kontrolatzeko ahalmena dute, bertara informazioa gehituz edo ezabatuz. Hau, aurrez aipaturiko *gate* edo ateek egiten dute posible. Azken finean, ate hauek *sigmoid* funtzio bat dira eta [0,1] bitarteko balioak har ditzakete.



2.11 Irudia: LSTM baten modulua.

Hasteko, lehen urratsa zer informazio ahaztuko dugun edo ez dugun erabiliko zehaztea izango da, eta hau ahazteko atea edo *forget gate* atearen bidez egingo da. Horretarako, x_t eta h_{t-1} bektoreak kateatu, eta dagokion W_f parametroekin biderkatuz, b_f atalasea gehitu eta ahazteko atea kalkulatuko da.

$$f_t = \sigma_t(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.6}$$

Ahazteko ateak 0 balioa badu, informaziorik ez dela pasako esan nahi du. 1 balioa hartzen badu, berriz, informazio guztia pasako da zelula egoerara.

Hurrengo pausua zelula egoeran zer informazio berri sartzen dugun zehaztea izango da, eta honek hainbat urrats ditu: lehenik, i_t sarrera atearen bidez zein informazio eguneratuko den erabakiko da. Ondoren, C_t balioak kalkulatuko dira, *tanh* geruzan. Balio berri hauek zelula egoerara gehitzeko hautagaiak izango dira:

$$i_t = \sigma_t(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.7}$$
$$C_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c) \tag{2.8}$$

Amaitzeko, bi bektore hauen arteko biderketa egingo da. Horrela, i_t sarrera ateko balioak pisu bezala erabiliko dira eta hautagai balioetako bakoitza zenbat eguneratuko den zehaztuko dute.

Azkenik, h_t uneko egoera ezkutua kalkulatu behar da zelularen irteera zehazteko. Horretarako, σ funtzioa kalkulatuko da, zelula egoerako ze balio pasako diren irteerara zehazteko. Honi tanh-tik pasatako zelula egoera berritua biderkatuko zaio, eta horrela irteera zehaztuko da.

$$o_t = \sigma_t(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.9}$$
$$h_t = o_t * tanh(c_t) \tag{2.10}$$

Gated Recurrent Units (GRU)

Gated Recurrent Units arkitektura [Cho et al., 2014] lanean aurkeztu zen eta *LSTM* sarearen aldaera bat baino ez da. Ahazteko atea eta sarrera atea ate bakar batean konbinatzen ditu, eguneraketa atea deitua, eta gainera, zelula eta ezkutuko egoerak ere bat egiten ditu biak. Azken finean, *LSTM*-aren bertsio sinplifikatu bat da.

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{2.11}$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2.12}$$
$$\widetilde{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{2.13}$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h}_t \tag{2.14}$$

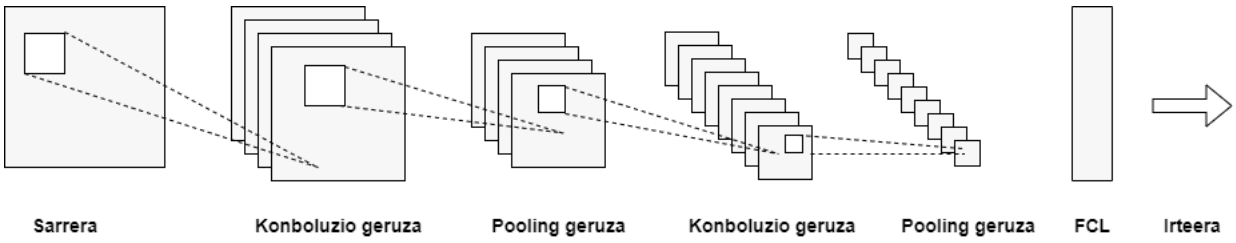
2.2.3 Sare konboluzionalak

Sare neuronal konboluzionala konputagailu bidezko ikusmenean sarri erabiltzen arkitektura da, irudien sailkapen eta ezagutzan esaterako, beste hainbat arlotan geroz eta aplika-

zio gehiago baditu ere. Sare hauek, gaur egun erabiltzen diren bezala, [LeCun et al., 1998] artikuluan azaldu ziren. Orduz geroztik, garapen nabarmena izan dute gaur egun arte.

Esan bezala, sare hauen puntu indartsua irudien analisia da. Gai dira irudietan ezaugarri desberdinak aurkitzeko, puntu, marra edo ertzak bezalako bereizgarri sinpleenetatik abiatu eta aurpegi edo bestelako objektuak aurkitzeraino.

Sare konboluzionalak txandakaturiko konboluzio eta *pooling* geruzez daude osaturik, (2.12 irudia). Amaieran, *fully connected layer* edo *FCL* bat ere izan ohi dute, sailkapena egiteaz arduratzen dena. Azken hau ordea, atazaren arabera alda daiteke.



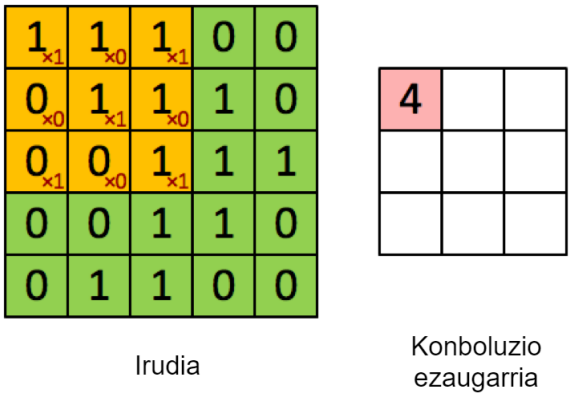
2.12 Irudia: Sare konboluzional baten egitura, non konboluzio eta *pooling* geruzak txandakatzen diren. Amaieran sailkapena egiteaz arduratzen den *Fully connected layer (FCL)* geruza.

Konboluzio geruza

Konboluzio geruza bat irudiaren gainean hainbat iragazki pasatzea bezala uler daiteke. Geruza honen sakonera irudiari aplikatuko zaion iragazki kopurua izango da, eta iragazki edo filtro bakoitza patroi desberdin bat identifikatzeko erabiltzen da. Beraz, geruza honetatik pasatzean, irudiaren tamaina txikituko da, sakonera izan ezik, iragazki kopuruaren arabera haziko baita.

Azpiko 2.13 irudian ikus daiteke, 3x3 tamainako filtro laranja irudiaren gainean pasatzen. Iragazki honek, irudiko pixel guztiak pasako ditu, eta posizio bakoitzean x_1 edo x_0 operazioa gauzatuko da irudiko pixelen gainean, irteeran guztien batura ezarriz. Beraz, irteerako ezaugarrien sakonera maila konboluzio geruzaren sakonera maila izango da.

Zehaztu beharreko beste bi parametro garrantzitsu iragazkiaren tamaina eta iragazkiaren mugimendua edo *stride* dira. Azken honek iragazkia zenbat mugituko den zehaztuko du pausu bakoitzean, eta hau zenbat eta handiagoa izan, irudiaren tamaina hare gehiago murriztuko da.



2.13 Irudia: Konboluzioa.

Iturria https://icecreamlabs.com/2018/08/19/3x3-convolution-filters%e2%80%8a-%e2%80%8aa-popular-choice/

Pooling geruza

Pooling geruzaren helburu nagusia irudiaren tamaina oraindik eta gehiago txikiagotzea da, bi helburu nagusi lortzeko: batetik, sarearen konexio maila txikiagotzea eta honen lana arintzea, eta bestetik, sarearen *overfitting* edo gehiegizko doitzea eragoztea.

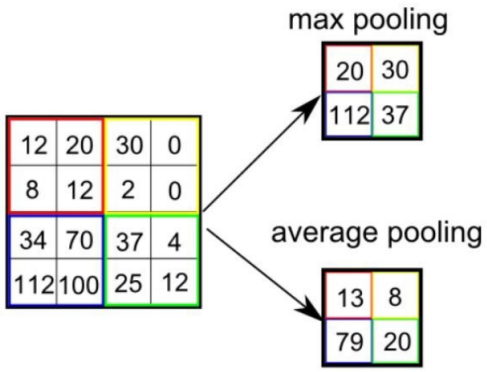
Bi pooling mota nagusi daude: *Average pooling* eta *Max pooling*. Biek helburu bera dute, baina heuren funtzionamendua zertxobait desberdina da. Azken finean, biek egiten dutena irudiko zati bat taldekatu, eta honen tamaina txikitzea da, horrela irudi osoa zeharkatuz.

2.14 irudiko adibidean, 2x2-ko *pooling* filtroa aplikatzen da. *Max pooling*-ek, taldekatzetik baliorik handiena duen elementua aukeratzen du. *Average pooling*-ek, aldiz, kasu honetan lau balioen batezbestekoa kalkulatu eta hau esleitzen du. Ikus daiteke, beraz, nola 8x8 tamainako irudi bat 2x2 tamainako irudia bihurtzen den bi kasuetan.

FCL geruza

Azken *fully connected layer* edo guztiz konektaturiko geruza, sailkapena egiteaz arduratzen den geruza da. Irudiko pixel bakoitza independenteki tratatuko da, eta sailkapen geruzaren bidez klase desberdinen gainean aukeratuko da.

Artearen egoeran, sare konboluzionalen aldaera desberdinak ere oso erabiliak dira, fun-



2.14 Irudia: Average pooling eta Max pooling.

Iturria: https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks

tzionamendu konplexuagoa dutenak. Horien artean, *R-CNN (Regions with CNN)*, *Fast R-CNN* eta geroago azalduko den eta proiektuaren parte izan den *Faster R-CNN*.

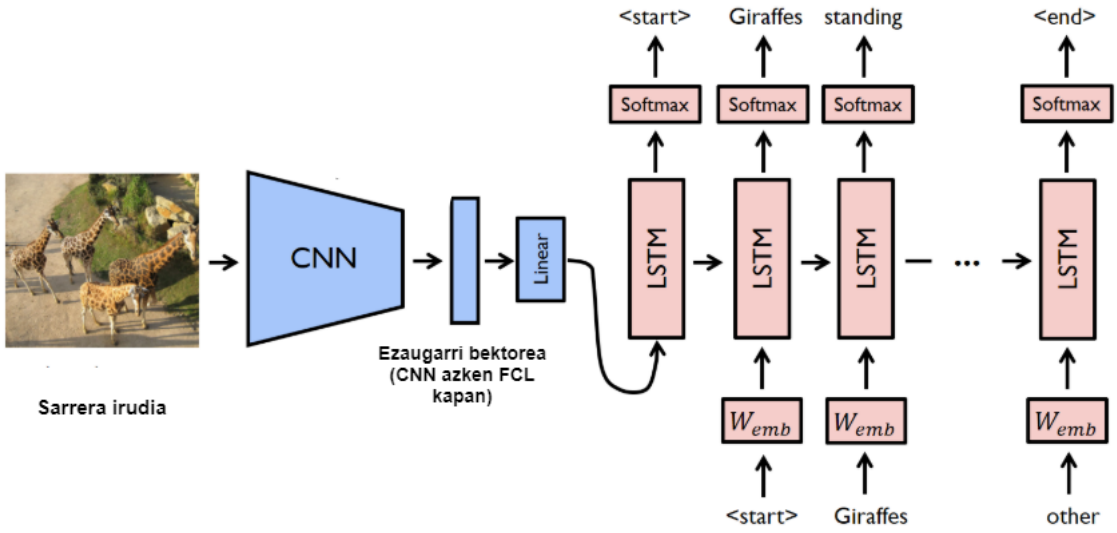
2.3 Image captioning arkitekturak

Testu bidezko irudien deskribapenak sortzeko arkitektura gehienetan, oinarrizko hezurdura bera jarraitzen da: kodetzaile - dekodetzaile edo ingelesez *encoder - decoder* arkitektura.

Egitura honen bidez, sarrera, luzera finko bat duen bektore batera mapatzen da kodetzailearen bidez, nolabait sarrerako datuak "kodetuz". Dekodetzaileak, berriz, bektore hauek hartu eta irteera balioa lortzeko beste mapaketa bat gauzatzen du, "kodetutako"informazio hori "dekodetuz".

Image captioning arloan asko erabiltzen den arkitektura da hau, [Vinyals et al., 2015] argitalpenean agertzen den gisan. *CNN* edo sare konboluzional baten bidez, irudien matrizeak sarearen sarrera izanik, irudi hauen ezaugarriak gordetzen dituzten bektoreak lortzen dira (2.15 irudia). Horrela, kodetzailearen eraginez sarrerako irudien bektore errepresentazioa eskuratzen da, eta hauek irudien ezaugarri desberdinak gordetzen dituzte.

Ondoren, ezaugarri bektore hau jaso eta 2.15 irudian ikus daitekeen bezala, *LSTM* dekodetzailetik pasatzen da. Dekodetzailean, ezaugarri bektorearen eta aurreko W_{emb} hitzaren



2.15 Irudia: *Image captioning* kodetzaile - dekodetzaile eredua. Sarrera bezala irudi bat emanda, irudiaren ezaugarri bektorea lortuko da *CNN*-tik (kodetzailea), eta bektore hau *LSTM*-aren sarrera izango da, ezaugarrietan oinarrituz irudiaren deskribapena sortu dezan (dekodetzailea).

Iturria: https://www.analyticsvidhya.com/blog/2018/04/solving-an-image-captioning-task-using-deep-learning/

arabera, pausu bakoitzean probabilitate distribuzioa kalkulatuko da hiztegiaren gainean *softmax* geruzan, eta probabilitaterik altuena duen hitza izango da esaldiko hurrengoa.

Hainbat token jada izendatuta egongo dira esaldiaren hasiera eta bukaera zehazteko: hasiera < *start* > tokenak zehaztuko du, eta esaldiaren amaiera < *end* > tokenak.

Jakina da, ordea, argazki bat deskribatzen ari garenean, zenbait objektu edo irudiaren zenbait ataletan gehiago zentratzen garela; hau da, gure deskribapeneko hitz bakoitza nolabait irudiko objektu edo atal baten gainean atentzio gehiago jarrita esango dugu. Hau dela eta, kodetzaile - dekodetzaile ereduari atentzio mekanismo bat gehitzen zaio, sarea y_t irteera bakoitzerako $(x_1, ..., x_n)$ sarrera sekuentziako atalik garrantzitsuenak identifikatzera behartuz ([Xu et al., 2015]).

Lehen atentzio mekanismoa [Bahdanau et al., 2014] artikuluan aurkeztu zen. Atentzio mekanismoaren helburua dekodetzailearen h_t uneko egoera ezkutua hartu eta sarrerako sekuentziako informazio garrantzitsua lortzen duen $\hat{v}_t$ testuinguru - bektorea lortzea da. Beraz h_t t uneko dekodetzailearen egoera ezkutua izanik, k irudiaren v_i ezaugarrientzat $\alpha_{i,t}$ atentzio-pisuen bektorea kalkulatzen da:

$$a_{i,t} = w_a^T tanh(W_{va} v_i + W_{ha} h_t) \tag{2.15}$$

$$\alpha_t = softmax(a_t) \tag{2.16}$$

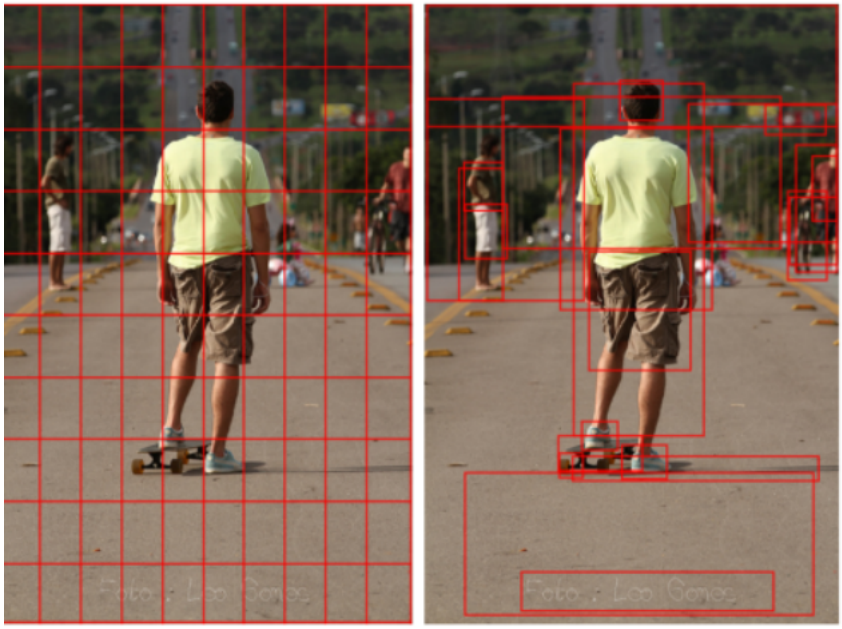
non W_{va}, W_{ha} eta w_a sarearen parametroak diren.

Azkenik, irudiaren ezaugarri bektorea eta atentzio-pisuen bektorea biderkatzen dira, ezaugarri bakoitzari dagokion pisua emanez, eta hau dekodetzaileko sarrera bezala erabiliko da, hitzaren embedding-arekin batera.

$$\hat{v}_t = \sum_{i=1}^{k} \alpha_{i,t} v_i \tag{2.17}$$

Atentzioa, beraz, *CNN*-tik ateratako irudien ezaugarri bektoreen gainean egiten da oro har. Ezaugarri hauek ateratzeko irudia uniformeki hainbat zati berdinetan banatzen da, irudiaren edukia edozein izanda ere (2.16 irudia), zati guztiek pisu berdina hartuz nolabait. Hala ere, erakutsi da atentzio hau irudiko objektu eta regio nabarien gainean ezartzean emaitza hobeak lor daitezkeela ([Anderson et al., 2018]). Esaterako, gizakiak irudi bat deskribatu behar duenean, irudiaren hainbat zatik zeresan handiagoa izan ohi dute, eta ez dute garrantzi bera irudiaren atal guztiek.

Hau horrela, irudiak kodetzeko objektu detekzio sistemak erabiltzen dira. Objektu bakoitza kutxa edo *bounding box* bidez identifikatzen da, 2.16 irudian ikus daitekeen bezala. Modu honetan, irudiko atalik esanguratsuenak lortzen dira, eta atentzioa honen gainean ezar daiteke irudiaren banaketa uniformearen gainean ezarri beharrean. Kasu honetan, irudi batean K objektu detektatzen badira, K bektore lortuko dira irudia kodetzean, bakoitzak objektu baten ezaugarriak gordeko dituelarik, eta hau izango da atentzio mekanismoaren sarrera.



2.16 Irudia: Irudiaren banaketa uniformea eta objektu detekzioa.

Iturria: [Anderson et al., 2018]

3. KAPITULUA

Arkitekturaren diseinua

Proiektu hau gauzatzeko bi *image captioning* arkitekturatan oinarritu naiz batez ere. Horietako bat [Xu et al., 2015] artikuluan oinarritzen den sistema bat izan da, baina zenbait desberdintasun ditu. Sistema hau probatu eta ulertu ondoren, hainat aldaketa burutu dira kodearen gainean, [Anderson et al., 2018] artikulua erreferentzia izanik, eta sistema honekin gauzatu da esperimentazio eta proben zatirik handiena, ondoren emaitzak alderatu ahal izateko.

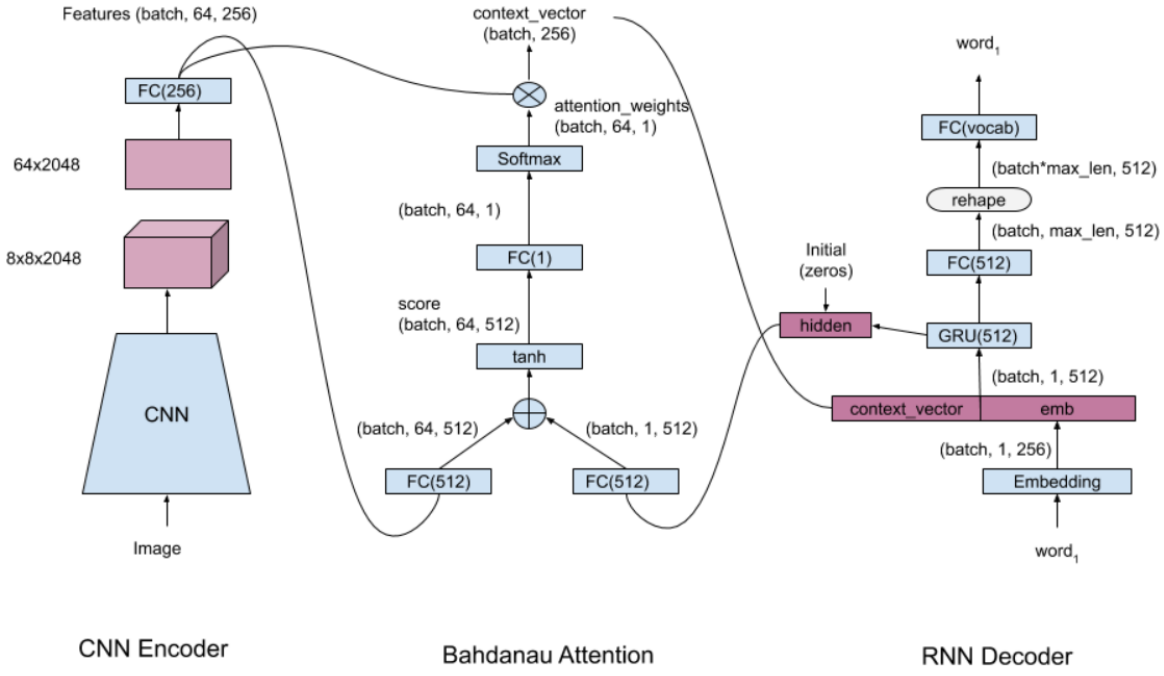
Jarraian 2.3 ataleko *encoder - decoder* egituran oinarrituriko bi sistema hauek azalduko dira modu zehatzago batean, eta heuren arteko desberdintasunak azalduko dira.

3.1 *Show, attend and tell*

Lehen arkitektura honetan, atentziodun kodetzaile - dekodetzaile sistema bat erabiltzen da, irudien ezaugarriak eskuratu eta hauen testu deskripzio automatikoa sortzeko. Arkitektura orokorra 3.1 irudian ikus daiteke zehatzago.

Irudien ezaugarriak lortzeko, *Inception V3* ([Szegedy et al., 2015]) izeneko arkitektura erabiltzen da. Objektu detekziorako eta irudien azterketarako asko erabiltzen den modeloa da *Inception v3*. Sistema hau, bloke simetriko eta asimetrikoez osaturik dago, horien artean, konboluzioak, *pooling*-ak, *dropout*-ak, *FCL* edo guztiz konektaturiko geruzak eta *batch* normalizazioak gauzatzen direlarik.

Proiektu honen kasuan, [Deng et al., 2009] artikuluan aurkezten den *ImageNet* irudi datu-



3.1 Irudia: *Show, attend and tell.* Lehen modeloaren arkitektura orokorra.

multzoan dago aurre-entrenatuta. Horrela, egin beharreko ataza bakarra gure datubaseko irudiak jada entrenaturiko modelo honetatik pasatzea da, horrela, irudi bakoitzari dagokion ezaugarri tensoreak lortuz. Irudi bakoitza saretik pasa aurretik, aurreprozesaketa prozesu batetik pasatzen dira: lehenik, irudi guztiei $299px$ *x* $299px$ tamaina ezartzen zaie eta ondoren, pixel guztien balioak normalizatzen dira balioa guztiak $[-1,1]$ balioen tartean egon daitezen. Behin pauso hauek emanda, datu-baseko irudi guztiak 64 tamainako sortetan antolatu eta *Inception v3*-tik pasatzen dira, *batch* bakoitzeko $64x8x8x2048$ tamainako tensoreak eskuratuz, non irudi bakoitzari $8x8x2048$-ko tensorea dagokion. Bektore hauek, ordea, lautu egiten dira, (*flatten* operazioa), $64x2048$ tamainara aldatuz, eta hauek dira ondoren erabiltzen direnak.

Atentzio mekanismo bezala aurretik 2.3 atalean azalduriko eta [Bahdanau et al., 2014] artikuluan aurkezturiko atentzio mekanismoa erabiltzen da. Honek bi sarrera izango ditu: aurretik aipaturiko kodetzailean lorturiko *batch*-eko irudien ezaugarri tensoreak eta dekodetzaileko *GRU* modeloaren ezkutuko egoera. Sarrera hauek emanik, testuingurubektorea kalkulatu eta hau dekodetzaileari pasatuko dio.

Dekodetzailea ere oinarrituriko artikuluan agertzen denaren desberdina da, eta *LSTM* modelo bat beharrean, 2.2.2 atalean azalduriko *GRU* arkitektura bat erabiltzen da. Hala ere,

funtzionamendua berdina da: testuinguru-bektorea eta aurreko hitzaren *embedding*-a batu, eta sare errekurrentetik pasako ditu, ondoren bi *FCL* igaro eta probabilitaterik altuena duen hitza itzultzeko.

Oinarritu naizen kode originala [1] *TensorFlow* tutorial bat izan da. Proiektuan zehar, kodeari hainbat aldaketa egin eta zenbait funtzionalitate edo aplikazio erantsi dizkiot:

- *test_step* urratsa gehitu: entrenamendu garaian, urrats hau gehitu da datu-multzoaren garapen atalaren gaineko errorea kalkulatzeko. Kode originalean, entrenamendu datu-multzoarentzat soilik kalkulatzen zen errorea, eta urrats honen bidez garapen atalaren eboluzioa ere aztertu ahal izan da. Gainera, errore hau eta entrenamendu atalarena grafiko berean batu dira, biak alderatu ahal izateko.

- *BLEU* kalkulua: Entrenamendu *epoch* bakoitzerako *BLEU-1*, *BLEU-2*, *BLEU-3* eta *BLEU-4* kalkulua ezarri da garapen datu-basearen gainean. Hau gauzatzeko, aurrez aipaturiko *test_step*-etaz baliatu naiz eta sorta bakoitzarentzat iragarritako hitzak gordetzen joan naiz, ondoren hauekin esaldiak osatu eta *BLEU* kalkulatu ahal izateko bakoitza bere 5 erreferentziazko esaldiekin alderatuz. Modu honetan, kalkulu hauen emaitzak ere grafikoan adierazi dira aurreko bi balioekin batera. Inplementazio garaian hainbat arazo direla eta, funtzioaren gaineko *@tf.function* ezabatu behar izan da, zeinak grafo egitura bat sortzen duen entrenamendua azkarragoa izan dadin.

- Grafikoak gorde: funtzio bat sortu da, iterazio edo *epoch* bakoitzean sorturiko grafikoa irudi gisan *Google Drive*-n gordetzeko.

- Deskribapenak gorde: entrenamendu bakoitzean 5 indize sortzen dira ausaz, eta indize hauei dagozkien irudiak ebaluatzen dira urrats bakoitzean heuren deskribapenak lortuz. Deskribapen hauek fitxategi batean idazten joaten da, ondoren analisi kualitatibo bat egitea ahalbidetuz *caption* hauen bilakaera aztertuz.

3.2 *Top-down bottom-up attention*

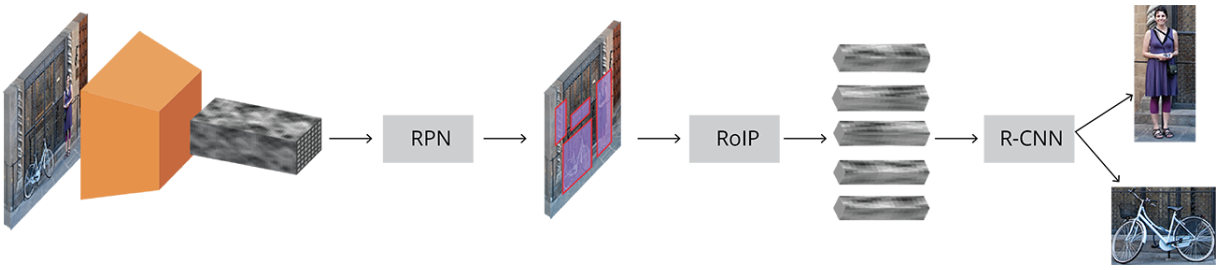
Aurretik aipaturiko arkitektura oinarritzat hartuta, zenbait aldaketa gauzatu dira, esan bezala, [Anderson et al., 2018] artikuluan oinarrituta. Aldaketarik aipagarriena kodetzailean egon da. Aurre-entrenaturiko *Inception V3* modeloa erabili beharrean, jarraian sakonki

[1] https://www.tensorflow.org/tutorials/text/image_captioning

azalduko den *Faster R-CNN* [2] objektu detekzioko sistema txertatu da, eta arkitektura orokorrera moldatu da, egin beharreko aldaketak eginez, horien artean, sarrera eta tensoreen formak moldatzea.

3.2.1 Faster R-CNN

Proiekturako erabili dudan kodetzailea [Ren et al., 2015] artikuluan aurkezturiko *Faster R-CNN* arkitektura izan da (3.2 irudia), gure kasuan *Visual Genome*-n aurre-entrenatua 1600 objektu desberdinetarako. *Faster R-CNN* objektu detekzioko sistema bat da. Irudi bat emanik, helburua *bounding box* edo objektu-kutxen zerrenda bat lortzea da, non bakoitzari etiketa bat esleituko zaion probabilitate jakin baten arabera. Horretarako, sistema hainbat atal desberdinetan banatuta dago:



3.2 Irudia: *Faster R-CNN* modeloaren arkitektura orokorra

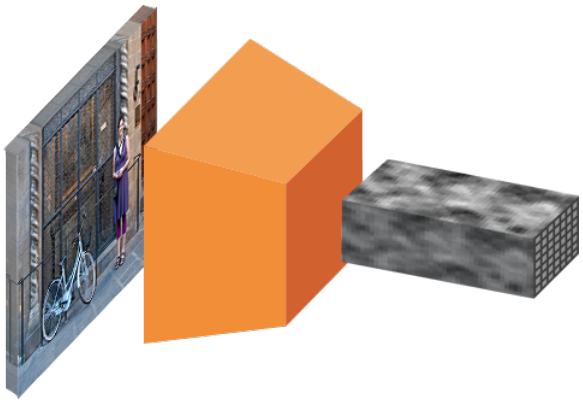
Iturria: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

Aurreprozesaketa

Irudiak *Altuera x Zabalera x Sakonera* tensore bezala errepresentatzen dira. Atal honetan, tensore hauek aurre-entrenaturiko *CNN* batetik pasatzen dira, eta tarteko geruza konbolu-zional bat erabiltzen da ezaugarri mapaketa lortzeko. Aurre-entrenaturiko sare konbolu-zional gisa *ResNet-101* eredua erabiltzen da erabilitako arkitekturan.

3.3 irudian ikus daitekeen bezala, sare konboluzionala irudiaren ezaugarriak ikasiz joango da, sinpleenetik hasi (ertzak, koloreak...) eta ezaugarri konplexuagoak lortu arte. Horrela, sakonera handiagoko tensore bat lortuko da, irudiaren ezaugarriak gordetzen dituena, eta hau hurrengo atalerako sarrera bezala erabiliko da.

[2] https://github.com/airsplay/py-bottom-up-attention/blob/master/demo/demo_feature_extraction_attr.ipynb



3.3 Irudia: Irudi baten mapaketa konboluzionala ezaugarriak lortzeko.

Iturria: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

Region proposal network (RPN)

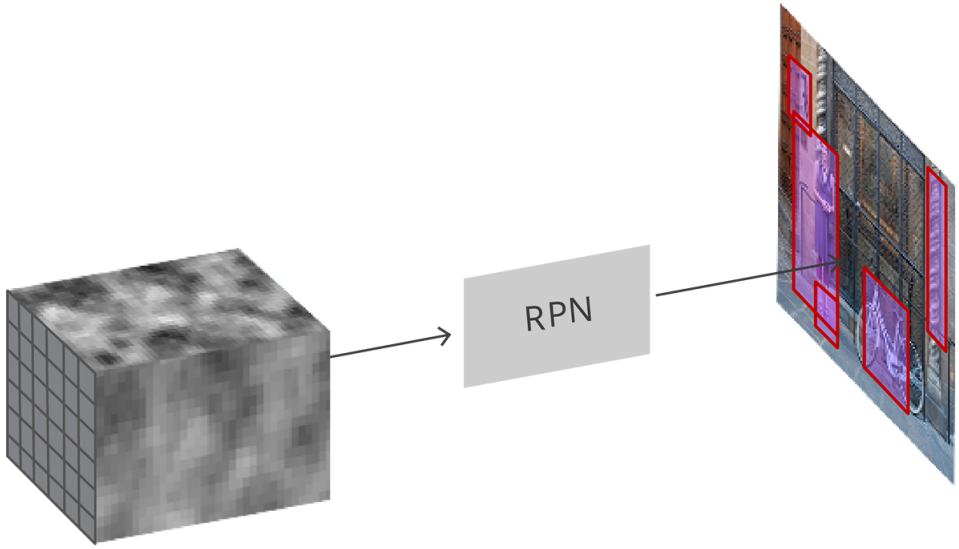
Hurrengo helburua, ezaugarri tensore horietatik abiatuz irudiko objektuak errepresentatuko dituzten *bounding box* edo kutxen proposamenak lortzea da (3.4 irudia), eta horretarako, *anchor* egituraren kontzeptua azaldu beharra dago aurrenik.

Anchor bat *bounding box* finko bat da azken finean. Irudian zehar, erdiguneak izango diren puntuak zehazten dira irudiaren tamainaren arabera, eta puntu hauen gainean tamaina eta neurri desberdineko *anchor*-ak zehazten dira, objektuen kokapenerako erreferentzia izango direlarik. Kutxa desberdin hauetako bakoitza $x_{zentroa}$, $y_{zentroa}$, *zabalera* eta *altuera* balioek zehaztuko dute.

Hau argiago geratzeko, 3.5 irudiko ezkerreko zatian ikus daiteke nola erdigune beraren gainean proportzio desberdineko hainbat *anchor* definitu daitezkeen. Erdian, irudiko puntu baten gaineko kutxa desberdinak ageri dira, eta azkenik, eskuinean, irudiko kutxa guztiak. Ikus daitekeen bezala, zentro puntu ugari definitzen dira irudian, proposamen kopurua ere handia izan dadin.

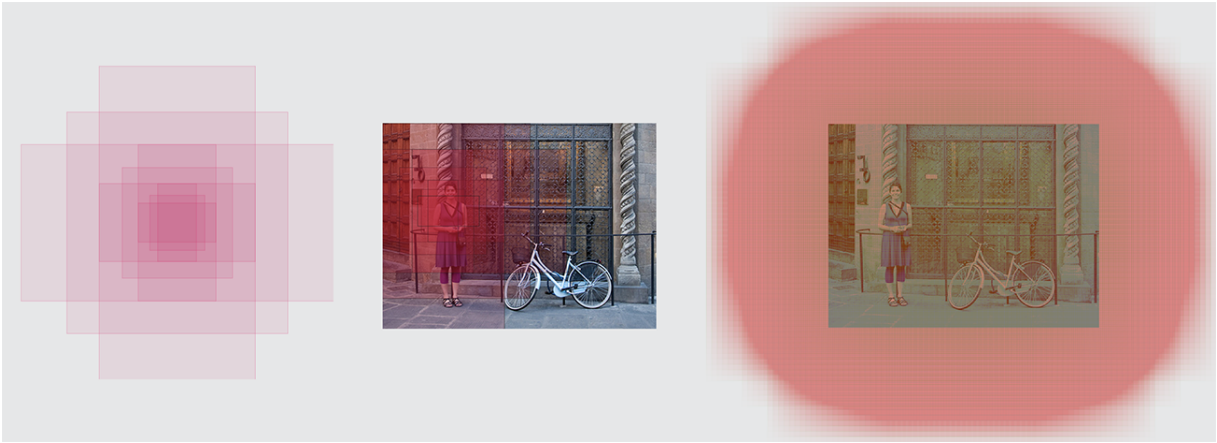
Behin hau zehaztuta, *RPN*-aren eginkizuna da proposamen kutxa hauek guztiak hartu eta objektuei dagozkien proposamenak lortzea. Horretarako, aurrez lorturiko ezaugarri mapaketak konboluzio geruza batetik igaroko dira, eta ondoren beste bi konboluzio geruza paralelo egongo dira.

Sailkapen geruzan *anchor* bakoitza objektua izateko probabilitatea kalkulatuko da. Probabilitatea oso baxua bada, ezin esango da objektu bat errepresentatzen duenik, eta *back-*



3.4 Irudia: *Region proposal network*

Iturria: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/



3.5 Irudia: Ezkerrean: zenbai *anchor* zentro berdinaren gainean. Erdian: Zenbait *anchor* irudiko puntu batean. Eskuinean: *anchor* guztiak.

Iturria: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

ground edo atzekaldea dela esango dugu.

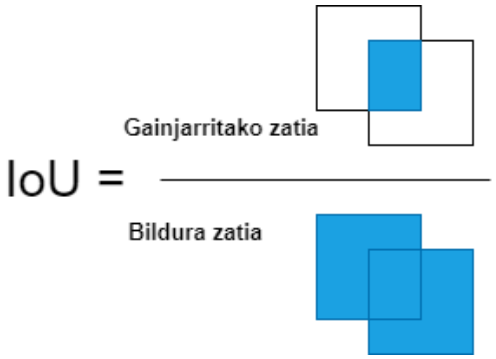
Erregresio geruzatik 4 balio iragartzen dira: $\Delta_{Xzentroa}, \Delta_{Yzentroa}, \Delta_{zabalera}$ eta $\Delta_{altuera}$. Hauek aurrez zehazturiko *anchor*-ei aplikatuko zaizkie doiketa txikiak egiteko.

Horrela, objektua izateko probabilitateekin eta jada doitutako erreferentzia kutxekin, objektuen proposamen kutxa multzo on bat lortzen da.

Hala ere, lortzen diren proposamenekin arazo bat gertatu ohi da. *Anchor* egiturak oro har gainjartzen diren bezala, objektuen proposamenak ere gainjarri egiten dira sarritan. Hau da, objektu bera adierazteko hainbat proposamen kutxa edo *bounding box* egoten dira. Errepikapen arazo hau konpontzeko, *Non-Maximum Suppression (NMS)* algoritmoa erabiltzen da.

Non-Maximum Suppression

Algoritmo honek proposamen guztiak ordenatzen ditu objektua izateko probabilitatearen arabera eta lista ordenatu honen gainean lan egiten du. *IoU (Intersection over Union)* (3.6 irudia) eragiketa kalkulatzen du bi proposamenen artean, eta emaitzak aurrez definituriko atalase edo muga bat gainditzen badu (0.6 gure kasuan), probabilitate txikiena zuen proposamena baztertu egiten da.



3.6 Irudia: *Intersection over Union eragiketa (IoU).*

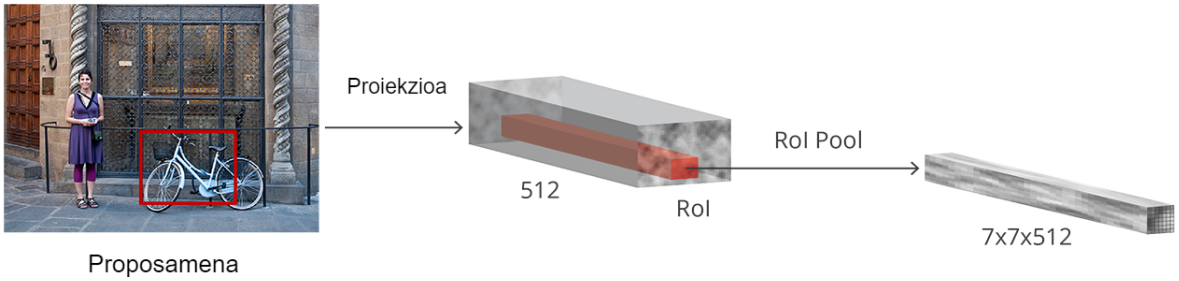
Modu honetan, proposamen kopurua aurrez zehazturiko kopuru batetara mugatzen da.

RoI pooling

RPN urratsaren ondoren, oraindik klaserik ez duten hainbat objektu proposamen izango ditugu. Hurrengo pausoa, *bounding box* hauek sailkatzea izago da dagokien klaseen

arabera.

Horretarako, aurrenik tarteko pauso bat burutu behar da. Jadanik lorturiko ezaugarri mapaketa konboluzionala erabiltzen da. Objektu proposamen bakoitzeko, irudiaren mapaketa murriztu bat proiektatzen da, 3.7 irudian bezala, eta ondoren, *RoI pooling* bitartez, proposamen bakoitzaren mapaketa lortzen da.



3.7 Irudia: *Region of Interest Pooling (RoI).*

Iturria: https://tryolabs.com/blog/2018/01/18/faster-r-cnn-down-the-rabbit-hole-of-modern-object-detection/

Objektu bakoitzaren mapaketaren dimentsioak $7x7x2048$ dira, baina hauek aldakorrak izan daitezke atazaren arabera. Honen helburua, konputazio kostua murriztea da, bai eta etiketa esleipena errazago egitea ere.

Region-based CNN (R-CNN)

Faster R-CNN arkitekturaren azkenengo urratsa da *R-CNN*. Aurreko atalean proposamen bakoitzaren ezaugarriak lorturik *RoI pooling* bidez, proposamen hauek sailkatzea soilik geratzen da dagokien etiketaren arabera.

Beraz, azken urrats honetan, proposamen bakoitzaren ezaugarri mapaketa bektore bihurtu eta guztiz konektaturiko (*FCL*) bi geruza pasako ditu, heuren *ReLU* aktibazioekin. Azkenik, beste bi *FCL*-etan banatuko da egitura, eta bakoitzak bere eginkizuna izango du:

Batetik, lehen sareak $N+1$ unitate izango ditu, N klase edo etiketa kopurua izanik. Azken klase hori objektu ez izateari edo atzekaldea izateari dagokio. Beraz, klase guztien gaineko probabilitate distribuzio bat kalkulatuko da *softmax* bidez, eta horrela, proposamenari dagokion klasea zehaztuko da.

Bestetik, $4N$ unitateko guztiz konektaturiko geruza bat. Klase posible bakoitzerako, *RPN* atalaren antzera, $\Delta_{Xzentroa}, \Delta_{Yzentroa}, \Delta_{zabalera}$ eta $\Delta_{altuera}$ kalkulatuko dira erregresio bidez, *bounding box*-ak bat etor daitezen detektaturiko objektuekin.

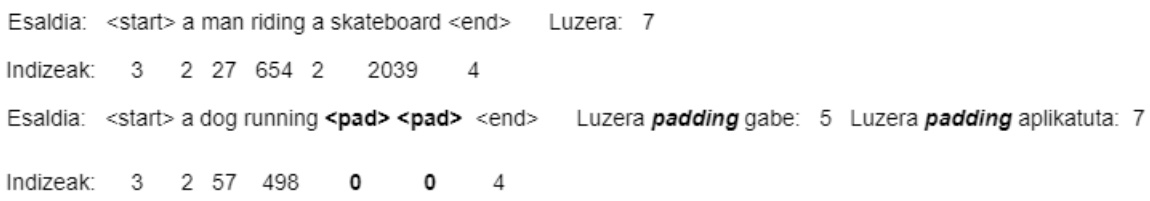
Amaitzeko, gure proiekturako irudi bakoitzerako 10 objekturekin geratu gara. Horretarako, 3.2.1 atalean azalduriko *NMS* algoritmoa erabili dugu, 0.2 balioko mugarekin, eta horrela, gainjartzen ez diren 10 objektu zehaztu irudi bakoitzean, bakoitza bere klasearekin, eta gure kasuan, atributu batekin klase bakoitzerako, modu berdinean lortua. Objektu bakoitza 2048 luzerako bektore batek zehazten du, eta irudi bakoitza, beraz, $10x2048$ tamainako tensore bat izango da.

Behin sistema hau azalduta, aurreko sistemarekin alderatzearren, orain sorta bakoitzeko $64x10x2048$ tamainako tensorea eskuratuko da, $64x64x2048$ beharrean. Hau kontuan hartuta, sistema forma hauetara egokitu da behar bezala lan egin dezan. Atentzio mekanismoa eta dekodetzailea berdin mantendu dira, baina esperimentazioa gauzatzerako orduan 3.4.2 ataleko teknika desberdinak probatzeko hainbat geruza gehitu dira probak egiteko.

3.3 Hiztegia

Irudien deskribapenak sortzeko eta hauekin lan egiteko beharrezkoa da hiztegi bat izatea. Kasu honetan, hiztegia entrenamendurako erabiliko diren deskribapenetatik lortu da. Hau da, deskribapen guztietan agertzen den hitz desberdin bakoitza tokenizatu eta hiztegira gehitzen da aurretik bertan ez badago. Token hauekin guztiekin hiztegia sortzen da, non token bakoitzari zenbaki edo indize bat dagokion. Badaude hainbat token berezi:

1. < start > : deskribapenaren hasiera adierazten du.
2. < end > : deskribapenaren amaiera adierazten du.
3. < unk > : iragarritako hitza hiztegian ez dagoela esan nahi du.
4. < pad > : sarea entrenatzeko dekodetzailearen sarrerako elementu guztiek luzera bera izatea ezinbestekoa da. Horretarako, deskribapen luzeenaren luzera hartu eta luzera horretara iristen ez diren esaldi guztiak 0-koz edo < pad > tokenez betetzen dira. Prozesu edo pauso honi *padding* deitzen zaio. Demagun azpiko kasuan (3.8 irudia), luzera maximoa 7 dela. Beraz, luzera hori baino txikiagoa duten esaldi guztiei *padding* aplikatuko zaie.



3.8 Irudia: *Padding* prozesuaren adibide bat.

Honetaz gain, entrenamendurako nahiz garapen faserako, esaldi guztietako hitzak letra xehez egongo dira idatzita. Gainera, *BLEU* kalkuluak egiterako orduan, erreferentzia esaldietako puntuazio marka guztiak ezabatu dira, < start > eta < end > tokenekin batera. Horrela, kalkulua batez ere aurresandako hitzen gainean egitea lortzen da, puntuazio atal edo letra larrien ondorioz penalizaziorik jaso gabe.

3.4 Entrenamendua

Atal honetan entrenamenduaren zenbait atal eta ataza zehaztuko dira. Batetik, entrenamendurako erabili diren kostu-funtzioa eta optimizazio algoritmoa zein izan diren eta, bestetik, entrenamendua modu eraginkor eta azkarrago batean egiteko erabili diren teknika desberdinak azalduko dira.

3.4.1 Kostu-funtzioa eta optimizazio algoritmoa

Proiektuko helburuak gauzatzeko erabili den **kostu-funtzioa** 2.1.1 atalean aipatu eta azalduriko entropia gurutzatua izan da. Kasu honetan, entropia gurutzatu kategorikoa erabili da zehazki, bi etiketa klase baino gehiago daudelako; kasu honetan, hiztegiko hitzak adina klase edo kategoria egongo dira.

Aurrerago agertu den bezala, erabili den *batch* edo sortaren tamaina 64-koa izan da; hau da, uneoro, irudi eta deskribapen bakarrarekin jardun beharrean, 64 irudiren deskribapena lortzen da paraleloan. Hori dela eta, dekodetzailetik igaro eta hainbat urratsen ondoren, $64x1$-eko bektorea lortuko da, non elementu bakoitza esaldi bakoitzerako iragarritako hurrengo hitza izango den. Errorea kalkulatzeko, erreferentziako esaldietan dagozkien hitzak hartu eta tamaina bereko bektore bat sortuko da, parametro bezala kostu funtzioari eman eta hauen arteko errorea kalkulatzeko.

Modu honetan, sorta bakoitzeko errorea kalkulatu ondoren **optimizazio** funtzioari deitzen zaio. Nire kasuan, 2.1.1 atalean zehazturiko *Adam* optimizatzailea erabili dut. Sortako deskribapen bakoitzeko iragarpen guztien gaineko errorea kalkulatu ondoren, optimizazio algoritmoari deituko zaio, eta dagozkion gradiente eguneraketak burutuko dira, sortaz-sorta prozesua errepikatuz.

Sorta guztiak, hau da, datu-multzoko irudi guztiak pasatzen direnean, *epoch* bat pasa dela esango da.

3.4.2 Entrenamendu eraginkortasuna hobetzeko teknikak

Gauzatutako esperimentazioan, entrenamenduaren eraginkortasuna hobetzeko hainbat teknika desberdin aplikatu dira, hauen arteko konbinaketa desberdinak probatuz. Estrategia hauek *teacher forcing, dropout, batch normalization* eta erregularizazioa izan dira.

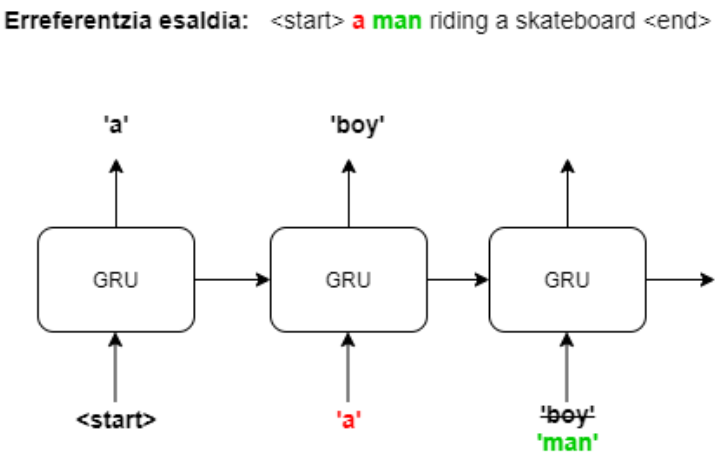
Teacher forcing

Teacher forcing testu-sortzaileak hobeto eta modu eraginkorrago batean entrenatzeko erabiltzen den teknika bat da.

Proiektu honetan, *GRU* arkitektura erabili da dekodetzaile bezala. Sare errekurrenteetan, oro har, denbora-une bakoitzean hitz bat aurresaten da irteeran, eta hurrengo denbora-unean sarrera bezala erabiltzen da sistemak lorturiko hitz hori. Metodo honen bidez, sistemak iragarritako hitza erabili beharrean, erreferentziatzat erabiltzen den deskribapenaren dagokion hitza erabiliko da sarrera gisa denbora-une bakoitzean (ikusi 3.9 irudia).

Horrela, hitz bakoitzaren iragarpenak ez du izango aurreko denbora-uneko errorearekin zerikusirik, eta sarea modu efizienteago batean entrenatu ahal izango da.

Behin hau azalduta errazagoa da *caption* edo deskribapenen sorkuntza azaltzea. Aipatu den bezala, sare errekurrenteak hitz bat aurresaten du denbora-une bakoitzean. Hitz hau lortzeko, *softmax* geruza bat erabiltzen da hiztegiaren gainean probabilitate distribuzio bat kalkulatzeko, eta probabilitate altuena duen hitza izango da sorturiko deskribapenaren hurrengo hitza. Hurrengo denbora-unean hitz hau erabiliko da sarrera bezala, eta horrela jarraituko du < *end* > tokena aurresaten duen arte. Token hau iragartzean, esaldia bukatutzat emango da.



3.9 Irudia: *Teacher Forcing* teknikaren adibidea. Entrenamenduan zehar bakarrik aplika daiteke.

Dropout

Dropout [Srivastava et al., 2014] artikuluan aurkezturiko erregularizazio metodo bat da. Teknika honen bidez sarearen *overfitting*-a sahiestea da helburua, modu horretan, sarearen orokortze gaitasuna handituz.

Horretarako, sarearen hainbat unitate baztertzen dira ausaz entrenamendu fasean. Hau da, neurona bakoitzak sarrera desberdinetako balioak jasoko ditu iterazio bakoitzean. Horrela, zailagoa egingo zaio nolabait jasotako balioetaz "fiatzea"eta pisuak bananduago mantenduko ditu. Honek neuronak datu multzoan espezializatzea zailagoa egingo du, sareak generalizazio gaitasun handiagoa lortuko duelarik.

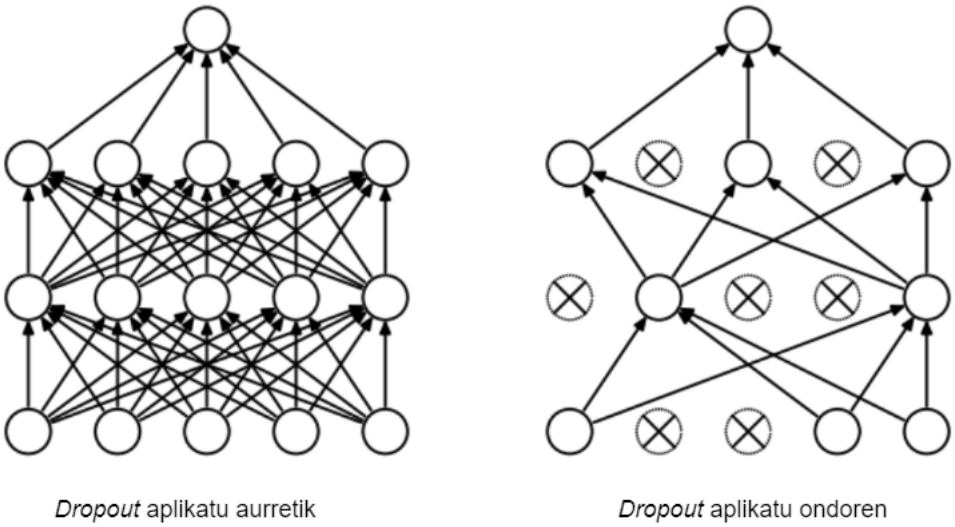
Jarraian datorren 3.10 irudian ikus daiteke nola hainbat neurona ez diren kontuan hartzen. Iterazio bakoitzean, ausaz aukeraturiko neurona desberdinekin egingo da hau, probabilitate baten arabera, eta esan bezala, iterazio bakoitzean sarrera desberdinak jasoko dituenez, pisuak orekatuago mantentzea lortuko da.

Proiektuan egindako probetarako, 0.2 balioa erabili da. Honek unitateen %20-a baztertuko dela esan nahi du.

Batch normalization

Batch normalization edo *batch norm*, ([Ioffe and Szegedy, 2015]) sare neuronalen abiadura, eraginkortasuna eta egonkortasuna hobetzeko erabiltzen den teknika bat da.

Batzuetan, entrenamendu datuen distribuzioa eta garapen edo *development* datuen dis-



3.10 Irudia: *Dropout* teknika erabiltzearen diferentzia.

Iturria: https://mc.ai/regularizando-nuestra-red-dropout/

tribuzioa ez da berdina izaten. Beste hitz batzuetan esanda, demagun $X \rightarrow Y$ mapaketa egiten duen sare bat izanik, X sarrera datuen distribuzioa zertxobait aldatzen dela. Kasu horretan, sarea berriro entrenatu beharko litzateke. Arazo honi *covariance shift* deritzo.

Hau konpontzeko *batch normalization* teknika erabili ohi da. Ezkutuko geruza baten kasuan, aurreko geruzako aktibazio balioak etengabe aldatzen direnez, aurrez aipaturiko arazoa gertatzen da. Metodo honen bidez, aktibazio balio hauek normalizatu egiten dira. Horretarako, *batch* bakoitzeko batezbestekoa eta desbideratze estandarra kalkulatzen dira.

Batch bakoitzak m elementu izanik, horrela kalkulatuko dira *batch* bakoitzaren μ_B batezbestekoa, σ_B^2 desbideratze estandarra, $\hat{x}_i$ normalizazioa eta y_i *batch normalization* irteera, non γ eta β prozesuan ikasten diren bi parametro diren:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{3.1}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{3.2}$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2}} \tag{3.3}$$

$$y_i = \gamma\hat{x}_i + \beta = BN_{\gamma,\beta}(x_i) \tag{3.4}$$

Modu honetan, berdin dio aurreko geruzako balioak zenbat aldatzen diren, izan ere, batezbesteko eta desbideratze estandarren balioak antzekoak izango dira teknika hau aplikatzen den geruzan. Beraz, aurreko geruzako balioak aldatuta ere, aldaketa hauek geruza honetan izango duten eragina murriztuko da nolabait, eta geruzako balioak egonkorragoak izango dira, hurrengo geruzek ere oinarri horretatik hobeto ikasiko dutelarik.

Erregularizazioa

Proiketu honetako esperimentazioan *L2* edo *Ridge* erregularizazio teknika erabili da, neurona sareak izan dezakeen *overfitting*-a sahiestu eta generalizazio gaitasuna handitzeko.

Metodo honen bidez penalizazio edo zigor bat esleitzen zaio kostu funtzioari. C penalizazioa honela kalkulatzen da, non n geruza kopurua, $w^{[j]}$, j geruzako parametroen matrizea, m sarrera kopurua eta λ erregularizazio parametroa diren:

$$C = \frac{\lambda}{2m}\left(\sum_{j=1}^{N} ||w^{[j]}||^2\right) \tag{3.5}$$

λ parametroak balio txikia hartzen duenean penalizazioa ere txikia izango da, C-ren balioa 0-tik gertu egongo baita. Erregularizazio parametroaren balioa handia bada, ordea, pisu handiak gehiago penalizatuko dira, helburua azken finean kostu-funtzioa minimizatzea baita.

Beraz, erregularizazio teknika honen bidez sareko parametroen balioak txikituko dira, batez ere balio handiak penalizatuz eta sarearen orokortze gaitasuna hobetuz.

3.5 Inplementazioa

Proiektuaren inplementazioa garatzeko bi liburutegi erabili dira. Orokorrean, erabili diren bi sistemak *Tensorflow 2.0*-n oinarrituta daude. *Tensorflow Google*-k garaturiko kode irekiko liburutegia da, adimen artifizial eta ikasketa automatikora bideratua. Sare neuronalak modu erraz batean erabili eta entrenatzea ahalbidetzen duenez, ikasketa sakoneko atazetan sarri erabiltzen den liburutegia da.

Hala ere, aipatu beharra dago, bigarren arkitekturako kodetzailea, hau da, *Faster R-CNN*-a, *PyTorch* liburutegian oinarriturik dagoela, izan ere, kodetzailea aparteko fitxategi batean dago inplementatua. *PyTorch* batez ere *Facebook*-ek garaturiko *Python* lengoaiarako liburutegia da, *Torch* liburutegian oinarrituta. *Tensorflow*-rekin batera, adimen artifizial eta ikasketa sakonean gehien erabiltzen den liburutegia da.

Hurrengo esteketan erabilitako arkitektura eta inplementazioen kodeak aurki daitezke:

- https://colab.research.google.com/drive/14otoafwEb4J8xbKSKFOfhElbLO9yQjYa?usp=sharing (Lehen eredua)
- https://colab.research.google.com/drive/1HgBnjVG3PDmm6RJqIJU2C1uKTvGFbH13?usp=sharing (*Faster R-CNN*)
- https://colab.research.google.com/drive/1wtxNGHUUYJnX6vYXpta88IvX_yJZwfxR?usp=sharing (Bigarren eredua)

4. KAPITULUA

Ebaluazioa eta emaitzak

4.1 Datu-multzoa

Proiektua gauzatzeko erabili diren datu guztiak *MSCOCO (Common Objects in Context)* datu-base publikotik lortu dira. Objektuen identifikazio, segmentazio eta deskribapenerako prestatuta dagoen datu-multzo handi bat da, eta irudien deskribapenak sortzeko bezalako atazetan sarri erabilia da.

Proiektu honetarako, *MSCOCO* datu-multzoaren bi azpimultzo erabili dira. Batetik, originala, non irudi eta deskribapen kopurua handia den. Kasu honetarako, ordea, handiegia suertatzen denez eta ezin denez tamaina horretako multzo bat entrenatu, azpimultzo txikiago bat sortu da eta hau erabili da esperimentazioaren atalik nagusienerako. Hala ere, datu-base originalaren gainean entrenamendu bat gauzatzea lortu da.

Datu-multzo originalak 138.000 irudi inguru ditu. Hauetatik 83.000 inguruk entrenamenduko zatia osatu dute, eta bakoitzak batezbeste 5 deskribapen izan ditu. Gainontzeko irudiekin, hots, 55.000 irudi inguru, garapen zatirako utzi dira. Hauetan, irudi bakoitzeko deskribapen kopurua txikiagoa izan da. Honetaz gain, *test* edo proba zatia osatzeko beste 5000 irudi erabili dira.

Sorturiko azpi-multzoak, aldiz, 25.000 irudi izango ditu, hauek ere bakoitzak 5 deskribapen izango dituelarik. Entrenamendurako, multzoa bi zatitan banatu da: *train* edo entrenamendu zatiak irudi guztien %80-a hartuko du, 20.400 irudi hain zuzen; *dev* edo garapen faseko multzoak, berriz, %20-a, hots, 5.400 irudi. Datu-multzo handiarekin bezala, 5000

irudi berak erabili dira *test* zatia osatzeko, eta irudi horien gainean *BLEU* metrika kalkulatzeko. Azken 5000 irudi hauek *MSCOCO* datu-multzoaren *dev* zatitik hartu dira.

4.2 Aurreprozesaketa

Entrenamendua hasi aurretik, arazoak ekidin eta guztia behar bezala joan dadin, aurreprozesaketa prozesu bat gauzatzen da, eta datu-multzoari, deskribapenei zehazki, hainbat aldaketa eragiten zaizkie.

Hasteko, deskribapen guztiak letra xehera pasa dira, eta esaldi guztiak tokenizatu eta banatu dira. Erreferentzia deskribapen hauetako hitzekin sortzen da hiztegia, 3.3 atalean zehatzago aipatzen den bezala.

Behin esaldiak letra xehean izanik, beste hainbat aldaketa gauzatzen dira ebaluaziorako, hurrengo 4.3 atalean xehetasunez azalduko den *BLEU* metrikaren kalkulua ahalik eta zehatzen gauzatzeko: batetik, puntuazio ikur guztiak ezabatu dira esaldi guztietatik; bestetik, hasiera eta amaierako <*start*> eta <*end*> tokenak ezabatu dira, eta gauza bera egingo da entrenamenduan sortzen diren deskribapenekin, baldintza berdinetan ebaluatzeko. Modu honetan, puntuazio ikur edo hizki larriek sor ditzaketen penalizazioak ekiditen dira.

4.3 Ebaluazio automatikoa: BLEU

Ereduak sorturiko esaldien eta erreferentziazkoen arteko antzekotasuna alderatzeko *BLEU (Bilingual Evaluation Understudy)* ebaluazio automatikorako metrika erabili da. Jarraian, honi buruzko azalpen zehatzago bat emango da.

BLEU([Papineni et al., 2002]) itzulpen automatikoan erabiltzen den ebaluazio metodo bat da. Itzulpen automatikoaz gain, ordea, testu sorkuntzarekin loturiko beste zenbait arlotan ere erabiltzen da, horien artean, *image captioning*. Metrika hau ondorengo printzipioan oinarritzen da: gure kasuan, sistemak sorturiko irudiaren deskripzioa emango zaio, eta honekin batera, irudiak dituen erreferentziazko deskribapenen zerrenda. Bi sarrera horiekin, sistemak sorturiko deskripzioaren eta erreferentzien arteko antzekotasuna kalkulatuko da. Erreferentzia esaldiaren eta modeloaren deskribapenaren arteko n-grama komun kopurua m izanik eta hautagai deskribapenaren n-gramak w_t, horrela kalkulatzen da bi esaldiren arteko antzekotasuna *BLEU*-n:

$$P_n = \frac{m}{w_t} \tag{4.1}$$

Hainbat erreferentzia esaldi egonik, guztietatik emaitza onena ematen duen balioa itzuliko du funtzioak.

1-grama komun kopurua aztertzea bada helburua, *BLEU1* kalkulatuko da; 2-gramekin, *BLEU2*; 3-gramekin *BLEU3*; 4-gramekin *BLEU4*, etab.

Hautagai deskribapenaren luzera oso laburra denean, penalizazio bat ezartzen da, hautagai honen puntuazioa ez dadin altua izan. Horrela, kalifikazio on bat lortzen duen hautagai baten eta erreferentzia deskribapenaren luzerak antzekoak izan beharko dute, eta hau hala ez denean, penalizazio bat ezarriko da. Beraz, *c* sistemak sorturiko deskribapenaren luzera izanik eta *r* erreferentziarena, honela kalkulatzen da *BP (Brevity Penalty)* penalizazioa:

$$BP = \begin{cases} 1 & baldin \quad c > r \\ \\ e^{(1-r/c)} & baldin \quad c \leq r \end{cases}$$

Penalizazio honekin, batezbesteko geometrikoa erabiltzen da azken *BLEU* balioa kalkulatzeko.

$$BLEU = BP \cdot exp(\sum_{n=1}^{N} w_n \log p_n)$$

4.4 Hiperparametroak

Esperimentazioan zehar, hainbat hiperparametro erabili dira jatorrizko emaitzak hobetu ahal izateko. Hauek izan dira erabili diren teknikak:

- *Dropout:* 0.0 eta 0.2
- *Batch* normalizazioa: geruza ezarri edo ez

- *L2* erregularizazioa: 0.00 edo 0.01
- Optimizatzailearen ikasketa-tasa: 0.001 edo 0.0001

Hiperparametro hauek *top-down bottom-up attention* ereduaren gainean eta datu-multzo txikiarekin soilik erabili ahal izan dira, denbora eta errekurtso falta direla eta. Hau horrela, parametro guzti hauen konbinaketa posible guztiak probatu dira eredu honetan, eta emaitza nahiz grafiko guztiak gorde, esplorazioa amaitzean hauen efektua alderatu ahal izateko.

4.5 Emaitzak

Atal honetan, bi ereduak ebaluatu ondoren lorturiko emaitzak zehaztuko dira, eta erabilitako hiperparametroen analisi bat egingo da. Gainera, lorturiko deskribapenen zenbait adibide ere emango dira.

Esperimentuak aurrez 4.1 atalean aipaturiko datu-multzo txikiaren gainean egin dira. Hauek izan dira datu-multzo murriztuarekin esperimentazioan zehar emaitzarik onena eman duten konfigurazioak:

Show, attend and tell:

- Hiztegiaren tamaina: 5000
- *Batch* edo sorta tamaina: 64
- *Embedding* geruzaren dimentsioa: 256
- *GRU*-aren egoera ezkutuaren dimentsioa: 512
- Optimizatzailearen ikasketa-tasa: 0.001
- Epoka kopurua: 15

Top-down bottom-up attention:

- Hiztegiaren tamaina: 5000
- *Batch* edo sorta tamaina: 64

- *Embedding* geruzaren dimentsioa: 256
- *GRU*-aren egoera ezkutuaren dimentsioa: 512
- Optimizatzailearen ikasketa-tasa: 0.0001
- *Dropout* balioa: 0.2
- Epoka kopurua: 15

Datu-multzo handiarekin ere lortu da entrenamendu bat gauzatzea. Hala ere, ezin izan da *test* fasearen gaineko emaitzik lortu, denbora eta errekurtso falta direla eta, baina balio izan du datu kantitatea handituz emaitzak nola alda daitezkeen ikusteko. Hona hemen erabilitako hiperparametroak:

Show, attend and tell:

- Hiztegiaren tamaina: 5000
- *Batch* edo sorta tamaina: 128
- *Embedding* geruzaren dimentsioa: 256
- *GRU*-aren egoera ezkutuaren dimentsioa: 512
- Optimizatzailearen ikasketa-tasa: 0.001
- Epoka kopurua: 30
- *Dropout* probabilitatea: 0.5

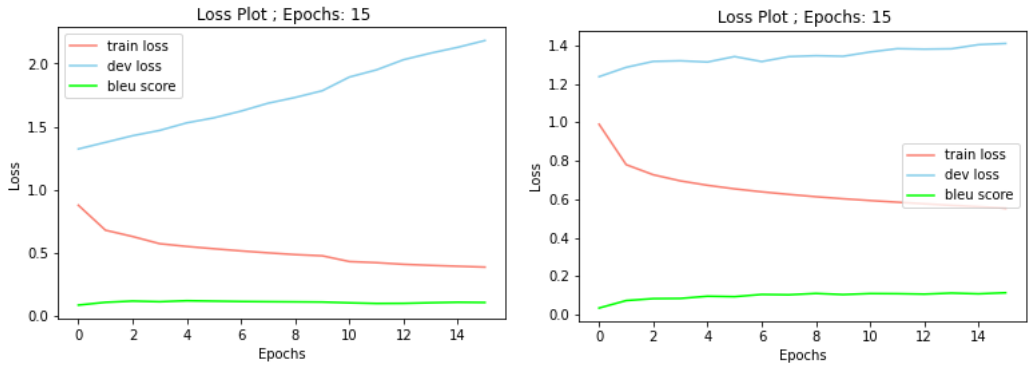
Esan bezala, bi kasuetarako datu-multzoa banatu egin da: *train*, *dev* eta *test* zatiak. Modu honetan, *dev* atala erabili da metrikaren balio onena lortzen zuen konfigurazioa identifikatzeko, eta azkenik *test* zatiarekin ereduak ebaluatu ahal izan dira.

Helburua bi ereduak baldintza berdintsuagoetan alderatzea zenez, esperimentazioan zehar datu-multzo murriztuarekin lorturiko emaitza onenak 4.1 taulan gordetzen dira, bi ereduen gainean: *show, attend and tell* eta *top-down bottom-up attention*. Bertan, sistema bakoitzarekin lorturiko garapen faseko *BLEU-1* eta *BLEU-4* eta *test* edo proba faseko *BLEU-1* eta *BLEU-4* gordetzen dira.

Hala eta guztiz ere, aipatu bezala, datu-multzo handiarekin entrenamendu bat gauzatu ahal izan da, eta garapen faseko *BLEU-1* eta *BLEU-4* kalkulatu ahal izan dira: 63.45 eta 18.54, hurrenez hurren.

Lorturiko emaitza hauei eta ondorioei dagokienez, ikus daitekeen lehen diferentzia nabarmena datu-multzo desberdinen arteko aldea da. Datu kopurua handitu ahala, emaitzek hobera egiten dutela ikusten da, beraz, etorkizuneko lan interesgarria izango litzateke datu kopurua handituz *top-down bottom-up attention* eredua ere entrenatzea eta emaitzak konparatzea. Honetaz gain, datu-base handiaren gaineko *test* faseko emaitzak lortzea ere egokia litzateke.

BLEU metrikaren balioaz gain, entrenamendu eta garapen faseko galera funtzioaren balioak (3.4.1) ere aztertu dira. Azpiko 4.1 irudian ikusten dira datu-multzo txikien gaineko entrenamendu nahiz garapen faseetako erroreen grafikoak. Nabarmena da, entrenamendu faseko erroreak behera egiten duen heinean, garapen edo *dev* faseko errorea gora doala, nahiz eta bigarren grafikoan ikasketa-tasa txikiagoa eta *dropout* erabiliz zertxobait egonkortzea lortu, eta ez da honen esperotako konbergentzia lortu. Hala eta guztiz ere, *BLEU* metrikaren balioa hobetzea lortu denez, gure kasuan ez da errorearen eta *BLEU* metrikaren arteko korrelaziorik aurkitu. Baliteke honen arrazoia errore edo *loss*-a irudi-deskribapen pare guztien batezbesteko bezala kalkulatzen delako izatea, eta *BLEU*, berriz, hauen guztien maximo bezala.



4.1 Irudia: Datu-multzo txikiekin lorturiko emaitza onenaren errore grafikoak bi ereduentzat.

Show, attend and tell ereduarekin, emaitzarik onenean lorturiko errorea 0.56 izan da, eta 1.52 garapen fasean. Bigarren modeloarekin, aldiz, 0.56 izan da entrenamendu ataleko errorea, eta 1.40 garapenekoa. Beraz, *top-down bottom-up attention* ereduak errore txikigoa izan duen arren *dev* fasean, *BLEU* metrikak emandako balioak zertxobait txikiagoak izan dira. Emaitza hauek, *test* fasean ere, islatu dira, eta hemen ere lehen ereduak lorturiko emaitzak pixka bat hobeak izan dira. Bestalde, inoiz ikusi gabeko irudien gaineko

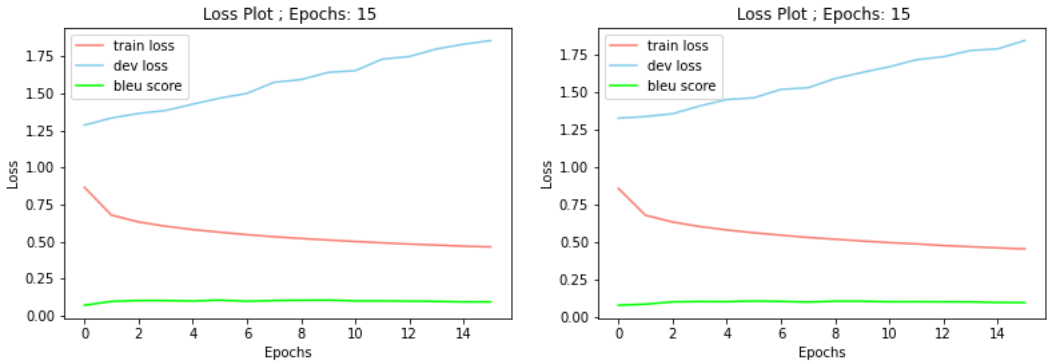
Eredua	Dev BLEU-1	Dev BLEU-4	Test BLEU-1	Test BLEU-4
Show, attend and tell	**59.32**	**11.62**	**57.48**	**11.15**
Top-down bottom-up attention	51.04	11.33	50.04	10.10

4.1 Taula: Datu-multzo txikian lortutako emaitzak.

ebaluazioa nahiko ona izan da, eta balioak garapenekoak baino zertxobait txikiagoak izan arren, bi ereduek ikasteko gaitasuna izan dutela esan daiteke.

4.5.1 Hiperparametroen eragina

Aipatu bezala, datu-multzo txikia erabiliz, hiperparametroen esplorazio bat gauzatu da *top-down bottom-up attention* ereduaren gainean. Balioak zehazturik, eginiko esplorazioan hiperparametro hauen konbinaketa guztiak probatu dira, eta ondorio orokor garbi bat atera da: kasu honetan behintzat, hiperparametroen eragina oso txikia eta ia hautemanezina izan da. 4.2 irudian ikus daitekeen adibidean bezala, errore balioak oso neurri txikian murriztu dira teknika desberdinak erabilita.

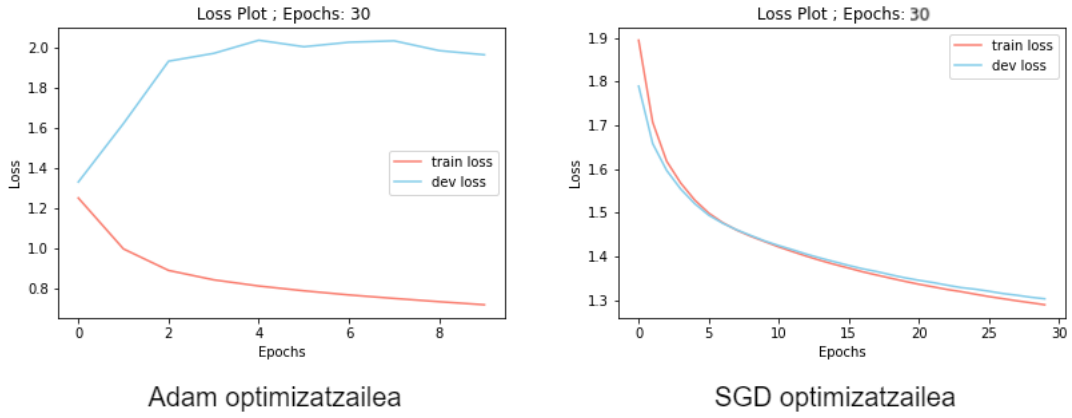


4.2 Irudia: Ezkerrean: ikasketa-tasa 0.001 ; Eskuinean: ikasketa-tasa 0.001 eta *batch* normalizazioa

Baliteke datu-multzoa nahiko txikia izanik, hauen eragina ere murriztu izatea faktore horrek, eta saiakera erakargarria litzateke multzo handiaren gainean ere esplorazio bat egiteko aukera balego, hiperparametroen balio desberdinak probatzea, hauen eragina handiagoa den edo ez ziurtatzeko. Bestetik, erabilitako hiperparametroen balioak egokienak ez izatea ere baliteke, eta hauek modu egokian kalibraturik ez egonik, izan dezaketen efektua eta eragina erabat murriztea.

Honetaz gain, esperimentazioaren hasieran optimizatzaile desberdinen arteko alderaketa bat ere egin zen: *Stochastic Gradient Descent (SGD)* eta *Adam*. 4.3 irudian, baldintza ber-

dinetan eginiko bi entrenamenduren grafikoak ageri dira. Lehen begiratuan, *SGD* erabiliz lorturiko emaitzak itxura hobea duela esan daiteke, eta bai entrenamendu nahiz garapen faseko erroreek konbergitu dutela ikusten da orokorrean errore balioa altua izanda ere. *Adam* erabiliz, aldiz, nahiz eta entrenamendu errorea jeistea lortu, *dev* multzoko erroreak gora egiten du, eta ez zen hau jeisterik lortu. Hau guztia kontuan izanik, hala eta guztiz ere *Adam* erabiliz lortzen ziren deskribapen edo *caption*-ak zentzuzkoagoak eta askoz egituratuagoak ziren, eta analisi kualitatibo baten ondoren, lorturiko deskribapenak errorearen balioan ez zirela islatzen ikusi zen.



4.3 Irudia: *Adam* optimizatzailearen eta *SGD* optimizatzailearen entrenamendu erroreen grafikoak

4.6 Eztabaida

Hurrengo 4.4, 4.5, 4.4 eta 4.7 irudietan, irudi berarentzat, datu-multzo txikiaren gainean entrenaturiko *show, attend and tell* eta *top-down bottom-up attention* ereduek lorturiko zenbait deskribapen ageri dira. Deskribapenak, esanguratsuak dira oro har eta irudien edukia ondo deskribatzen dute modu orokor batean. *BLEU* emaitzetan islatzen den bezala, deskribapenen egitura edo zenbait hitz desberdinak izanik ere, bi ereduek sorturiko esaldiak nahiko egokiak direla esan daiteke. Irudietarako, 5 erreferentzia esaldietatik bakar bat hautatu da erreferentzia gisa, eta ikus daiteke hauekin alderatuz sorturiko esaldiak zentzuzkoak direla.

Bi ereduek sorturiko deskribapenak eta lorturiko emaitzak ikusita, hauen analisiak hainbat ondorio ateratzeko aukera eman du, eta emaitzen zergatia zein izan daitekeen zenbait es-

Erreferentzia: `"a big commercial plane parked near some trucks in the snow"`

Show, attend and tell: `"a large plane is parked on a runway"`

Top-down bottom-up attention: `"a white large airplane parked on the back of a parking lot"`

4.4 Irudia: 1. Adibidea; Erreferentzia irudia eta bi ereduekin lorturiko deskribapenak.

pekulazio egiteko. Izan ere, oinarrituriko artikuluetan *top-down bottom-up attention* ereduaren emaitzak eta *BLEU* metrikaren balioak *show, attend and tell* arkitekturakoak baino hobeak direla frogatzen da, bat bestaren hobekuntza bat dela. Gure kasuan, ordea, bigarren eredu honekin lorturiko emaitzak zertxobait hobeak izan dira.

Hau hainbat arrazoi direla eta izan daitekeela uste da: batetik, esperimentazioan erabilitako irudi kopurua artikuluetakoa baino askoz txikiagoa izan da, eta segur aski honek eragina izan dezake; honetaz gain, *top-down bottom-up attention* ereduko *Faster R-CNN* kodetzaileak 36 objektu identifikatzen ditu [Anderson et al., 2018] artikuluan; gure kasuan, kopuru hau 10 objektura jeitsi da, batez ere, objektuen gainjartze eta errepikapena sahiesteko. Azkenik, baliteke dekodetzailean ere diferentzia egotea: erabilitako eredua artikuluan oinarriturik egonda ere, dekodetzailea desberdina da, eta baliteke faktore honek ere diferentzia eragin izana emaitzak lortzerako orduan.

Azkenik, esperimentazioan zehar beste faktore baten garrantziaz jabetzeko aukera ere izan da. Izan ere, jatorrizko kodean erabiltzen zen *sampling* estrategia aldatuta, hau da, dekodetzailean hiztegiaren gaineko probabilitate distribuzioa lortzean, hurrengo hitza aukeratzeko erabiltzen den estrategiaren arabera alde nabarmena ikusi da *BLEU* metrikaren

Erreferentzia: "a man riding a motorcycle down a curvy road"

Show, attend and tell: "a person on a dirt bike around a forest"

Top-down bottom-up attention: "a man riding a motorcycle on a curvy road"

4.5 Irudia: 2. Adibidea; Erreferentzia irudia eta bi ereduekin lorturiko deskribapenak.

Erreferentzia: "a zebra standing still in the grass near a tree"

Show, attend and tell: "a zebra standing in a field in area"

Top-down bottom-up attention: "a zebra is standing in a zoo"

4.6 Irudia: 3. Adibidea; Erreferentzia irudia eta bi ereduekin lorturiko deskribapenak.

balioan.

Erreferentzia: "a tall brick clock tower with giant bells"

Show, attend and tell: "a large clock tower with a clock on it"

Top-down bottom-up attention: "a clock tower with a clock on the top"

4.7 Irudia: 4. Adibidea; Erreferentzia irudia eta bi ereduekin lorturiko deskribapenak.

Oinarrituriko kodean hautaketa *kategorikoa* erabili da. Metodo honek, ez du probabilitate maximoa duen hitza hautatzen hurrengo hitz bezala. Kasu honetan, distribuzio guztia hartzen da kontuan, baina probabilitate handiagoa duten hitzek aukera handiagoa izango dute hautatuak izateko. Honen helburua, ikasketa eta entrenamenduan "zarata"pixka bat sortzea da, eta ereduaren orokortasuna hobetzea.

Hala ere, teknika honekin *BLEU* balioak oso baxuak zirela ikusi zen. Beraz, honen ordez, probabilitate distribuzioa kalkulatzean, probabilitate handieneko hitza izan da aukeratu dena. Horrela, aurreko emaitzak hiru edo lau bider hobetzea lortu da, eta kasu honetarako estrategia hau egokiagoa dela ikusi da.

Adibide batez azaltzearren, demagun gure hiztegian hurrengo 4.8 irudiko lau hitzak ditugula. Estrategia kategorikoa erabiliz, 5 proba egin ondoren ikus daiteke probabilitate handiena duen hitza dela gehien iragartzen dena; kasu honetan, *txakur* hitza 0.8-ko probabilitatez. Hala ere, gainontzeko hitzak ere hautatuak izateko aukera dago. Hautaketa maximoa erabiliz, ordea, aurresandako hurrengo hitza beti probabilitate handiena duena izango da.

Balioa	"kaixo"	"agur"	"txakur"	"zer"
Probabilitatea	0.2	0.3	0.8	0.6

```
Kategorikoa: ["txakur", "txakur", "zer", "agur", "txakur" ]
Maximoa: ["txakur", "txakur", "txakur", "txakur", "txakur" ]
```

4.8 Irudia: *Sampling* estrategia: *Categorical* eta *Argmax*

Honen arrazoia datu kantitatea txikia izatea izan daitekeela uste da. Izan ere, baliteke zenbait kasutan probabilitateak oso antzekoak izatea, eta ikasteko datu gutxi izanik, iragarritako hitza esperotakoaren oso desberdina bada, datuetatik ikasteko zailtasunak izan ditzake ereduak.

5. KAPITULUA

Ondorioak eta etorkizunerako lana

5.1 Ondorioak

Atal honetan proiektuaren eta eginiko lanaren ondorioak nahiz honek eragindako ondorio pertsonalak azalduko dira.

5.1.1 Proiektuaren ondorioak

Orokorrean, proiektuaren helburuak bete dira, nahiz eta zenbait ataza burutzea kosta egin den. Kontzeptu teorikoak neureganatzetik hasita, bi ereduak probatu eta aztertu dira, hauen emaitzak konparatuz. Lorturiko emaitzak onak izan dira datu-multzoa txikia izan dela kontuan hartuz, eta datu kopurua handituz emaitzek gora egiten dutela ikusi da. Hala ere, *top-down bottom-up attention* eredua *show, attend and tell* modeloaren hobekuntza bat dela jakinik, *BLEU* metrikaren balio baxuagoak eman ditu, eta honen arrazoiak zein izan daitezkeen aztertu da. Jarraian, proiektuaren ondorio nagusiak azpimarratuko dira.

Hasteko, bi ereduekin lan egiteko eta hauen funtzionamendua behar bezala ulertzeko, oinarrizko kontzeptu teorikoak sakonki ulertu eta landu dira. Prozesuan kontzeptu berri ugari nireganatu ditut, eta beste hainbat errepasatzeko aukera izan dut.

Gainera, testu bidezko deskribapenak sortzeko ikasketa sakon bidezko bi sistemak definitu dira eta prozesu honetan, bi kodetzaileen informazio errepresentazioaren azterketa egin da. Horrela, bi ereduak alderatu ahal izan dira, heuren desberdintasun eta portaerak iku-

siz. Hau gauzatu ahal izateko, atazarako egokia den *MSCOCO* datu-multzoa identifikatu da, eta honen tamaina mugatu da entrenamenduak gauzatu ahal izateko.

Honi loturik, hiperparametro desberdinen azterketa ere egin da [Anderson et al., 2018] artikuluan oinarrituriko *top-down bottom-up attention* ereduaren gainean. Horretarako, teknika eta parametro desberdinak erabili dira, eta hauek emaitzetan duten aldea aztertu da. Hala ere, azpimarratzekoa da esperimentazio garaian zenbait muga topatu direla: batetik, *show, attend and tell* ereduak lortzen duen irudien errepresentazioen dimentsioak beste sistemarenak baino handiagoak direnez, entrenamendu denbora oso luzea izan da eta honek esperimentu kopurua mugatu du; bestetik, probak egiteko eta kodea exekutatzeko *Google Colaboratory* plataforma erabili da, eta bertan, *GPU*-aren erabilera mugatuta dago. Arazo honek ere proba gehiago egitea eta esperimentu fasea arinagoa izatea eragotzi du.

Azkenik, aipatu *Google Colaboratory* plataforma horrelako proiektuetarako lan egiteko oso eroso eta egokia izan arren, datu-multzo handiagoekin lan egiteko arazoak ekar ditzakela. Hala eta guztiz ere, oro har proiektuaren hasieran zehazturiko helburuak betetzea lortu da. Ereduen arteko alderaketa egin da, eta hauen funtzionamendua ulertuaz kontzeptu ugari ikasi ditut.

5.1.2 Ondorio pertsonalak

Pertsonalki, proiektuaren garapena oso aberasgarri eta lagungarria izan da aspektu guztietan: eskuraturiko ezagutza eta honen aplikazioa, proiektuan sorturiko desbiderapen eta arazo guztiei aurre egitea eta etorkizunerako motibazioa.

Proiektuan lanean hasi aurretik, ez nuen inongo esperientziarik ia sare neuronal eta *image captioning* arloan. Hasteko, gaiari buruz informatzen eta irakurtzen hasi nintzen, eta erabiltzen ziren oinarrizko eredu nahiz egiturak aztertzen: sare neuronal konboluzionalak, sare errekurrenteak, *embedding*-ak, etab. Pixkanaka, kontzeptu hauek nireganatu eta egitura konplexuagoetara pasatu nintzen, eta proiektua amaitzean, "ikasketa-kurba"oso handia izan dela konturatu naiz.

Aipatu beharrekoa da, ezagutza honen barruan, nire programazio gaitasunak ere hobera egin duela. Orain arte *python* lengoaia ezagutzen banuen ere, ez nengoen inoiz *PyTorch* eta *TensorFlow* bezalako liburutegiak erabilita, eta ohartu naiz sare neuronal eta adimen artifizialaren munduan ezinbestekoak direla. Beraz, *image captioning* sistemen egitura konplexuaren ondorioz, liburutegien ezagutza asko handitu eta sakondu da.

Honetaz gain, orain arte graduan zehar emandako irakasgaiek ere eragina izan dutela ohartu naiz, eta praktikan jartzeko aukera izan dut. Esaterako, Datu Meatzaritza eta Algoritmoen Diseinua irakasgaietan ikusitako zenbait kontzeptu, edota sare neuronalen oinarrian dauden kontzeptu matematikoak ulertzeko Estatistika, Kalkulua eta Aljebra. Azkenik Proiektuen Kudeaketa irakasgaia ere lagungarria izan da proiektuaren plangintza eta jarraipena gauzatzeko.

Memoria idazterako garaian, zailtasunak aurkitu ditut batzuetan ez nengoelako ziur zer eta nola idatzi. Modu honetan, honek duen garrantziaz ohartu naiz; hau da, memoria edota dokumentu bat ongi eta txukun idazteak eta esan nahi dudana ongi azaldu eta adieraztearen garrantziaz. Horretarako, idatzi aurretik ideiak argi izatea ezinbestekoa dela uste dut, eta idatzi aurretik informazio hori antolatu eta egituratzeak asko lagun diezadakedala uste dut etorkizun batean.

Proiektuan zehar sorturiko arazoei dagokienez, hauei esker asko ikasi dudala esango nuke. Izan ere, aurreikusi gabeko hainbat arazo sortzean, hauen irtenbidea bilatu behar izan dut. Prozesu horretan, informazioa aurkitzea, informazio hau nola eta non aurkitu ikastea, eta ondoren honen aplikazio desberdinak gauzatzea, oso lagungarria suertatu zait eta askotan, momentuan bertan zail ikusi arren, buruhauste hauei aurre egiteko gai naizela ikusi dut.

Amaitzeko, ondorio orokor bezala, proiektua eta honek ekarri duen guztia oso positiboa izan da. Gauza ugari ikasteko aukera paregabea izan da, pertsonalki hazteko. Honetaz gain, ikasketa sakonak eta honek dakarkien guztia gustuko egin zait, eta landuriko ataza zehatz honek izan ditazkeen aplikazio desberdin eta aberasgarriak ikusita, adar honi buruz gehiago jakiteko jakinmina daukat. Adimen artifizial eta ikasketa sakonarekiko interes handia piztu zait, eta alde horretatik, ingurune honetan ikasten jarraitzeko motibatuta nago.

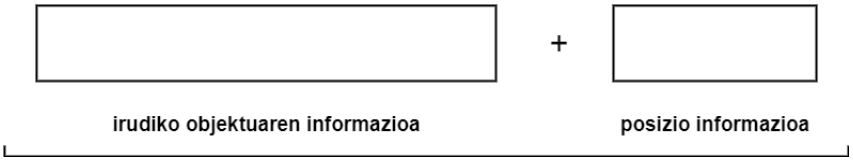
5.2 Etorkizunerako lana

Proiektuaren ondorioak aztertu ondoren eta eginiko lana ikusita, zenbait hobekuntza eta etorkizuneko lan interesgarri identifika daitezke, proiektua osoagoa egiteaz gain, lorturiko emaitzak hobetzeko. Atal honetan, etorkizunerako zenbait lan eta hobekuntza zehazten dira:

- **Emaitza eta esperimentazioaren hobekuntza:** Aurrerago aipatu den arren, proiektuan zeharreko eragozpenen ondorioz, hainbat hobekuntza egiteko utzi dira es-

perimentazioaren atalean: hasteko, *show, attend and tell* ereduarekin datu-multzo handiaren gainean entrenamendu bat egin bada ere, ez da *test* zatiaren gainean emaitzik lortzeko aukerarik izan, eta hau interesgarria izan liteke, modu honetan, konparaketa osoago bat egin ahal izateko; honetaz gain, interesgarria izango litzateke *top-down bottom-up attention* artikuluan oinarrituriko ereduarekin ere emaitzak lortzea datu-multzo handiaren gainean. Azkenik, *GPU* bat beharko litzatekeen arren, datu-multzo handiaren gainean ere hiperparametroen esplorazio bat egiteko aukera egongo balitz, lorturiko emaitzak eta oinarrituriko [Anderson et al., 2018] eta [Xu et al., 2015] artikuluetako emaitzak konparatzeko moduan egongo ginateke. Modu honetan, hiperparametroen eragina ikusi ahalko litzateke datu kantitatea handitzean.

- **Irudien errepresentazioa hobetu:** Irudiaren inguruko informazio gehiago izan dezake sare neuronalak, *top-down bottom-up* ereduak lorturiko irudi errepresentazioari objektu bakoitzaren posizioari buruzko informazioa gehitzea aberasgarria izan liteke, 5.1 Irudian agertzen den bezala:



5.1 Irudia: Posizioaren informazioaren gehikuntza objektu errepresentazio bakoitzari.

Horrela, irudian identifikaturiko 10 objektuei posizioari buruzko informazioa gehitu eta honen eragina aztertu ahalko litzateke. Horrela, baliteke irudiaren egitura eta oro har objektuen arteko posizio erlazioa hobeto ulertzea, heuren arteko posizio erlatiboa identifikatuz.

- **Objektu detekzioaren hobekuntza:** Analisi kualitatibo bat eginez, proiektuan zehar ikusi ahal izan da irudi batzuetan objektu berarentzat errepresentazio bat baino gehiago lortzen direla: hau da, objektu bera behin baino gehiagotan detektatzen du, eta hauen *bounding box*-ak gainjarri egiten dira. Honek zarata sor dezake eta irudiaren ulermenean zailtasunak ekarri. Beraz, nahiz eta proiektuan zehar lortu den puntu hau hobetzea objektu detektagialuaren parametroak aldatuz, arazoa sakonago azter litekeela uste dut.

Eranskinak

A. ERANSKINA

Proiektuaren helburuen dokumentua

A.1 Proiektuaren deskribapena eta helburuak

Proiektu honen helburua testu bidezko irudien deskribapen automatikoa sortzeko erabiltzen diren sare neuronaletan oinarrituriko bi sistema ulertu, sakonki aztertu, eta hauen arteko konparaketa egitea da. Horretarako, beharrezkoa izango da oinarri teorikoa sendotu eta arkitekuren gainean zenbait aldaketa gauzatzea.

A.2 Proiketuaren plangintza

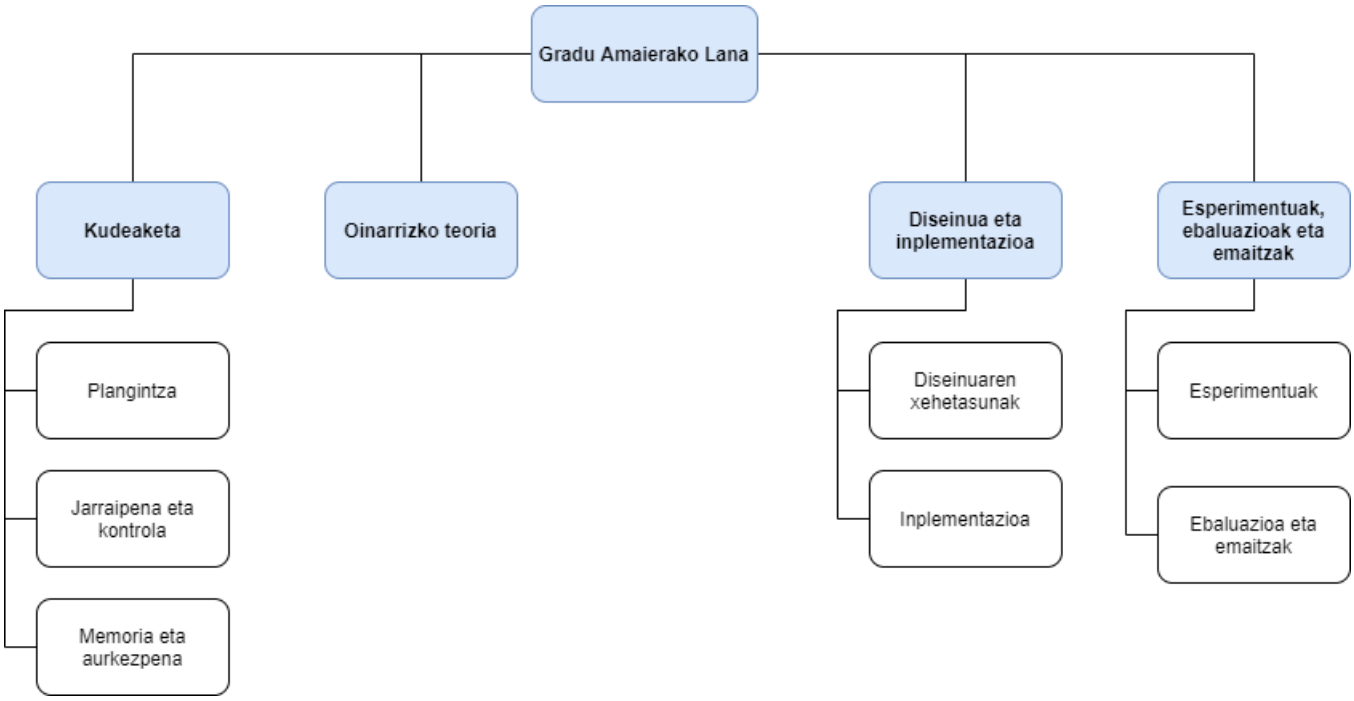
A.2.1 LDE diagrama

Proiektuan zeharreko lana banatu eta deskoponposatu da Lanaren Deskonposaketa Egitura (LDE) diagrama bidez (A.1 irudia).

A.2.2 Lan-paketeak

Plangintza

Atal honetan proiektuaren plangintza burutu da. Horretarako, proiektuaren helburuak, burutu beharreko atazak, emangarri zein mugarriak eta lan-metodologia zehaztu eta defini-



A.1 Irudia: Proiektuaren plangintza

tuko dira. Honetaz gain, proiektuaren arrisku eta aurrekarien analisi bat ere egingo da.

Jarraipena eta kontrola

Jarraipena eta kontrola proiektuaren prozesu osoan zehar kudeatuko den atal bat izango da. Bertan, hasierako plangintza eta ezarritako mugarriak betetzen direla bermatuko da. Edozein desbiderapenen aurrean, proiektuak aurrera diharduela ziurtatuko da, eta uneoro desbiderapen hauek jasoko dira. Hau guztia aurrera eramateko beharrezkoa izango da jarraipen bilera batzuk antolatzea.

Memoria eta aurkezpena

Hemen proiektuaren nondik norakoak azalduko dituzten memoria eta aurkezpena egingo dira:

- Memoria: Proiektuaren xehetasun guztiak gordeko dituen dokumentua.
- Aurkezpena: Proiektuaren defentsan lagungarri suertatuko den aurkezpena gauzatuko da. Bertan, proiektuaren atalik garrantzitsuenak azalduko dira modu argi eta antolatuan.

Oinarrizko teoria

Atal honetan proiektua gauzatzeko beharrezkoak diren oinarrizko kontzeptu teorikoak landuko dira, sakon ulertzeko asmoz. Egitura sinpleenetik hasita, testu bidezko irudien deskribapen automatikoa sortzeko erabiltzen diren arkitektura konplexuenetara.

Diseinuaren xehetasunak

Erabiliko diren bi *image captioning* sistemen xehetasun guztiak landuko dira.

Inplementazioa

Zati honetan, bi sistemak erabili ahal izateko inplementazio eta aldaketak txertatuko dira. Batez ere, lehen sistemaren gainean funtzionalitate berriak inplementatuko dira, eta aldaketak ezarriko dira bigarren sistemarekin lan egin ahal izateko.

Esperimentuak

Bi sistemen arteko desberdintasun eta diferentziak ikusteko beharrezko esperimentuak egingo dira, bai eta hiperparametroen esplorazio bat ere.

Ebaluazioa eta emaitzak

Esperimentuak ebaluatu eta emaitzen analisi bat egingo da. Horrela, gauzaturiko lanaren ondorioak ere aztertuko dira.

A.2.3 Emangarriak

Proiektuan zehar ondorengo emangarriak garatu behar dira:

- Kodea
- Memoria
- Aurkezpena

Lan-Paketea	Iraupena
Kudeaketa	**140**
Plangintza	10
Jarraipena eta kontrola	30
Memoria eta aurkezpena	100
Oinarrizko teoria	**80**
Diseinua eta inplementazioa	**150**
Diseinuaren xehetasunak	50
Inplementazioa	100
Esperimentua, ebaluazioak eta emaitzak	**100**
Esperimentuak	80
Ebaluazioak eta emaitzak	20
GUZTIRA	**470**

A.1 Taula: Lan-pakete bakoitzari esleituriko denbora

A.2.4 Mugarriak

Azpiko A.2 taulan proiektuko emangarriak entregatu beharreko datak zehaztuko dira:

Emangarria	Entregatze-data
Kodea	2020 / 06 / 21
Memoria	2020 / 06 / 21
Aurkezpena	2020 / 06 / 27

A.2 Taula: Mugarriak

A.2.5 *Gantt* diagrama

Atal honetan A.2 irudian agertzen den *Gantt* diagrama garatu da proiektuko atazen hasiera eta bukaera datak zehaztuz.

A.3 Lan metodologia

Proiektua garatzeko eta aurrera eramateko *Google Colaboratory* erabiliko da, eta honek neurona sareekin lan egiteko *GPU* txartelak erabiltzeko aukera emango du.

Lan-paketea		Hasiera	Bukaera	2020						
	Plangintza			Urtarrila	Otsaila	Martxoa	Apirila	Maiatza	Ekaina	Uztaila
Kudeaketa	Jarraipena eta kontrola	22/01/2020	10/07/2020							
	Memoria, aurkezpena eta posterra	13/05/2020	10/07/2020							
Oinarrizko teoria		24/01/2020	12/02/2020							
Diseinua eta inplementazioa	Diseinuaren xehetasunak	07/02/2020	19/02/2020							
	Inplementazioa	17/02/2020	20/05/2020							
Esperimentuak, ebaluazioa eta emaitzak	Esperimentuak	23/04/2020	10/06/2020							
	Ebaluazioa eta emaitzak	13/05/2020	10/06/2020							

A.2 Irudia: *Gantt* diagrama.

A.3.1 Bilerak

Ikasle eta tutoreen artean bilera bat gauzatuko da astero, zehazki, asteazkenetan 12:00etan. Edozein arazo izanez gero, aurrez hitz egingo da hitzordua aldatzeko aukera izateko. Bileretan, aurreko astetik eginiko lana aztertu eta hurrengo asterako helburuak zehaztuko dira, horrela proiektuari jarraitutasuna emanez.

COVID-19aren ondorioz, hasiera batean bilerak presentzialak izan arren, bideokonferentziaz egin behar izan dira martxotik aurrera.

A.3.2 Planifikatutako ordutegiak

Ikasleak ez du aurredefinituriko ordutegirik izango, baina errutina bat mantentzen saiatuko da eta planifikazio bat ematen.

A.4 Bideragarritasuna

Behin proiektua aurrera eramateko atalak zehaztuta, ezinbestekoa izango da honen bideragarritasuna bermatzea. Horretarako, ondorengo puntuak zehaztuko dira:

- Baliabideen kostua: Proiektuan zehar erabiliko diren baliabide guztien kostua doako dela eta hauek guztiak eskuragarri daudela bermatuko da.

- Baliabideen funtzionamendu bermea: Baliabide hauek guztiek behar bezala funtzionatuko dutela eta proiektuaren garapenerako eskuragarri egongo direla ziurtatuko da.
- Denbora: Proiektua aurrera eramateko nahikoa denbora izango dela bermatuko da, egon litezkeen desbiderapen eta ustekabekoek ekar ditzaketen atzerapenak kontuan izanik.
- Ikasle eta tutoreen arteko komunikazioa: Esan bezala, komunikazio hau eraginkorra eta erraza dela bermatuko da, bi zentzuetan edozein zalantza edo mezu ahalik eta azkarren erantzunez.

A.5 Arriskuak eta prebentzioa

Proiektuaren garapenean ohikoa da arrisku eta ustekabekoak egotea, eta proiektuaren nondik norakoa zeharo alda dezakete. Arrazoi hau dela eta, oso garrantzitsua da proiektua hasi aurretik arrisku hauek identifikatzea, eta egon daitezkeen arazoak ekiditeko neurriak hartzea. Hala ere, zerrenda honetan ez dagoen beste arriskuren bat egon daitekeela kontuan izan behar da.

A.5.1 Arriskuak

- Mota honetako proiektuetan oso ohikoa da desbiderapen handiak izatea aurrez egindako planifikazioarekin alderatuta.
- Prozesuan zehar beharrezko datuak galtzeko aukera egon liteke: kodea, grafikoak, emaitzen taulak, etab.
- Ikasketa sakoneko eredu bat entrenatu behar denez eta proba ugari egin, denbora asko beharko da entrenatzeko. Gainera, *Google Colaboratory* plataforman *GPU*-aren erabilera mugatua da, beraz, hau kontuan izan beharko da.

A.5.2 Prebentzioa

Aipaturiko arriskuei eta orokorrean gerta daitezkeen buruhausteei aurre egiteko, ondorengo prebentzio-plana sortu da:

- Plangintza sortzerako orduan kontuan hartu da desbiderapenak egoteko arriskua. Hori dela eta, plangintza malguagoa sortu da, moldagarriagoa izateko.
- Fitxategien galera sahiesteko, kode nahiz emaitza guztiak disko gogorrean gordetzeaz gain, *Google Drive* plataforman kopiak egin dira. Horrela, ordenagailuarekin edozein arazo izanez gero plataforman eskuragarri egongo dira.
- *GPU* erabilera mugari nahiz entrenamenduek hartutako denbora luzeari etekina ateratzeko, beste ataza batzuetan lan egingo da bitartean lana aurreratuz.

Bibliografia

[Anderson et al., 2018] Anderson, P., He, X., Buehler, C., Teney, D., Johnson, M., Gould, S., and Zhang, L. (2018). Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6077–6086.

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR, abs/1409.0473*, pages 6077–6086.

[Bottou, 2010] Bottou, L. (2010). Large-Scale Machine Learning with Stochastic Gradient Descent. *COMPSTAT*, pages 177–186.

[Cho et al., 2014] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. *CoRR, abs/1406.1078.*

[Deng et al., 2009] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. *IEEE*.

[Farhadi et al., 2010] Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J., and Forsyth, D. (2010). Every picture tells a story: Generating sentences from images. *European Conference on Computer Vision*, pages 15–29.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, pages Volume 9, Issue 8, 1735–1780.

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arVix*.

[Kingma and Ba, 2015] Kingma, D. P. and Ba, J. L. (2015). ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. *ICLR*.

[Kojima et al., 2002] Kojima, A., Tamura, T., and Fukunaga, K. (2002). Natural language description of human activities from video images based on concept hierarchy of actions. *International of Computer Vision*, pages 171–184.

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *IEEE (Volume: 86 , Issue: 11)*, pages 2278 – 2324.

[Papineni et al., 2002] Papineni, K., Roukos, S., Ward, T., , and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. *ACL*, pages 311–318.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*.

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, pages 323–533.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, page 15(56):1929−1958.

[Szegedy et al., 2015] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. (2015). Rethinking the inception architecture for computer vision. *cs.CV*.

[Vinyals et al., 2015] Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015). Show and Tell: A Neural Image Caption Generator. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164.

[Werbos, 1990] Werbos, P. J. (1990). Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, pages 1550–1560.

[Xu et al., 2015] Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zeme, R. S., and Bengio., Y. (2015). Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *JMLR: WCP volume 37*.