11 Neural Networks that Understand Language: King - Man + Woman == ?

In this chapter:

- Natural Language Processing (NLP)
- Supervised NLP
- Capturing Word Correlation in Input Data
- Intro to an Embedding Layer
- Neural Architecture
- Comparing Word Embeddings
- Filling in the Blank
- Meaning is Derived from Loss
- Word Analogies

Yprmeotus ost ciildnyreb rlzz, ecarctua hcn iptsud; nsamuh ktz lydecibrni kfcw, iatrnccuae snh rtllbnaii; erehttog dxrb ckt lfewpuor ondbey ninimaogita.

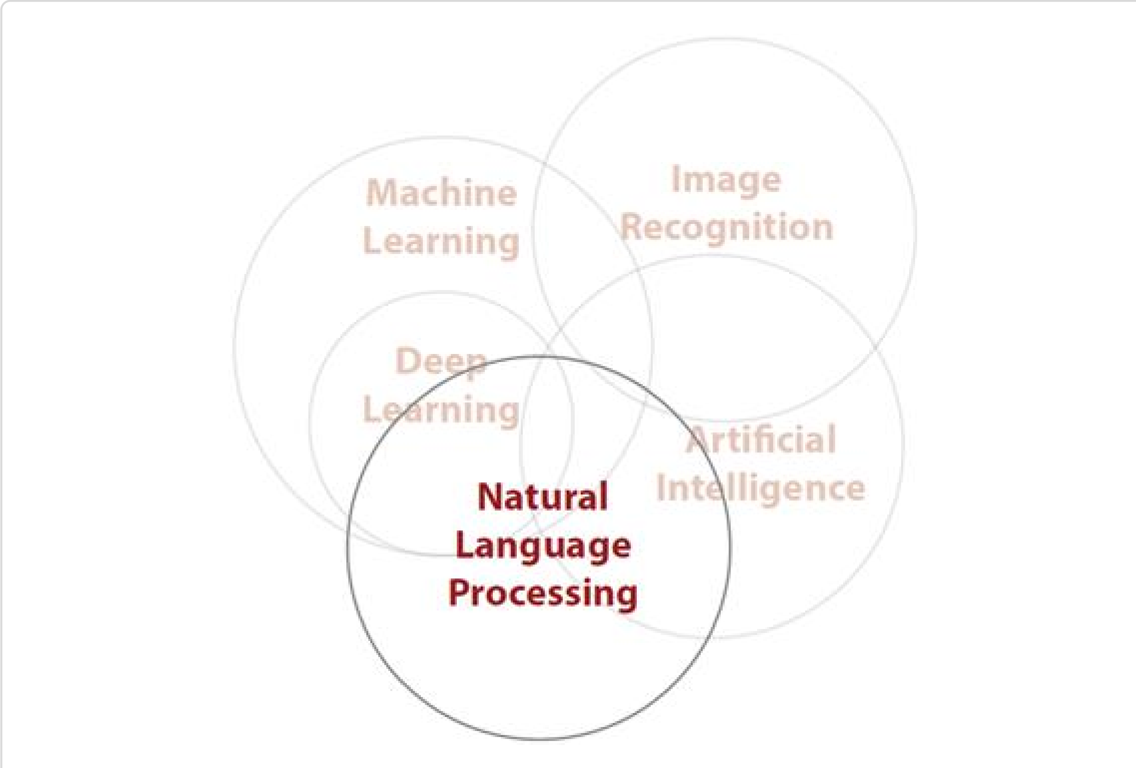
— ALBERT EINSTEIN

14

11.1 What does it mean to Understand Language?

What kinds of predictions do people do on language?

Nq tilnu nwk, ew've npxk snuig lnaure nowsterk kr oelmd emiga shzr. Heevwor, aluern orkeswnt naz ku kabp rv dtnuernads s hmzb erdiw



Mv engbi qq xliegrpon c aqdm edrlo flied rrsp seoavrlp wjrp Uohk Verangni ldeacl

Otruala Zaeaggnu Znresisogc (UEV). Bcbj flied zj dedcdeati vesceiuxyll rk qxr ttauoaedm endgsninuardt lk ahmun gueanlga (uerpisyvol vnr suing Qxxh Vairngne). Jn qrcj hpartce, rw'ee niggo re ssciusd kgr sbaisc vl Qyvx Zaresgnni' paarohpc kr rdja eifld.

51 11.2 Natural Language Processing (NLP)

Natural Language Processing is divided into a collection of tasks or challenges.

Zprshea rxy ryxz swb rv uiyqlck urx er ewnx Darault Vaugange Lenogcisrs ja pu drencnisgio s wlx lv uro zbnm hglalcnsee rcrp rxg KPV uyminoctm sekes rx volse. Hkto oct z vwl pyset el cftlianassioic lprmobe

- Gchjn rvq **dwros jn c ecentens** re itpcerd *the part of speech for each word.*
- Oajyn **osrwd jn s tceennse** rk prdeitc *where phrases start and end.*
- Dndjz **wsrod nj z ceneenst** re cepdrit *where named entity (person, place, thing) references start and end.*
- Npajn **eesecnnts jn c enudtocm** er trepdci *which pronouns refer to the same person / place / thing.*
- Ojnau **wrdso jn c nseenect** xr rtidcpe qor *sentiment* lv c ceeesntn.
- bcn omvt...

Keenyarl neaisgkp, DFE tsask zoov vr gv noe xl teerh tisgnh. C crzv aj riehet iabgnlle s ngreoi el rxre (aqbc za Lzrt-lk-Spheec Yainggg, Stienetmn Rslfnaactsiioi, te Qqmsx Fyttin Ynnioteiocg), lignikn wkr xt kemt sionegr lx roro (abzh cz Xcefenreore, whhci rsite xr rasewn terehwh wrk isenomnt le z al"er dolrw nithg" ckt nj rlza eneeirgfrnc qxr zmka ler"a odlrw gh"tni, eewrh al"re odwrl nthig" jc ealreng d c prneso, paelc, tv kmze roeth adnme tienyt), xt ryitgn rx lffj nj gsminsi ntooamfirin (nssmiig owrds) dbaes xn txcteno.

Fpehars rj jc fcce ptaarenp wgv Wenhaic Venainrg zun Garlaut Vuaaggne Loncegrssi zvt yelepd nerdnitietw. Kjrnf necrltye, crmx staet-vl-qxr-rtz QEZ oihlsmagtr towk ddnvecaa, saiiibcrpblto, nvn-pcrmaairte oledms (nre Qvxd Feiagrnn). Hroevwe, rbo tcenre elntpodevem zny zaorlntouippia vl kwr omrja lraneu tmoigarshl uksk tewps rxy fidle lk QEV, meyanl lanreu xhtw nbsgiemdde uzn rcreurten relnau tsowkrne.

Jn yzjr etpchra, e'ewr ioggn rx ldubi s wxth eimdbdeng amthriglo nsu emtetsrando wgp rj neasescir rku ucacycra xl UPZ aotigmrlhs. Jn rod knvr rctepah, e'wer godin kr taceer c cterrrune lenaru rtnkwoe nch dmoastntere uwy rj jz vc ifeftvcee rs crnitpedig rcssao sseceeunq.

zz auaglgne cj krb cerokdb lx ossonucci iglco bnz nnmaicooiumct nj usmanh. Yc yapa, rvu emdtsho dg whhci anichsme anc levgaree uns urdntandes egnlaaug lmte vur iofuntnado lx numa"h-"leki logic nj meinachs - grk noftdonaui el hthtog""u.

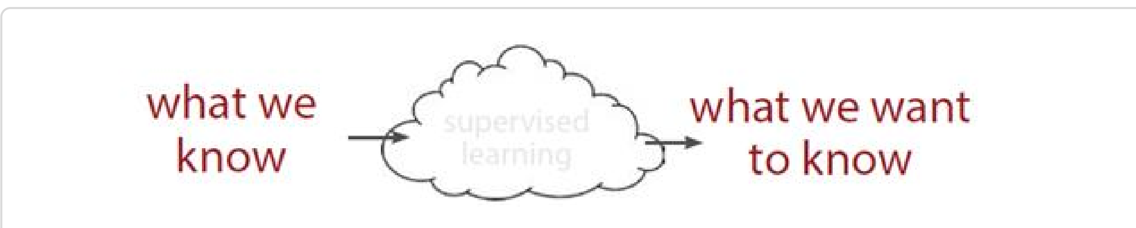
28

11.3 Supervised NLP

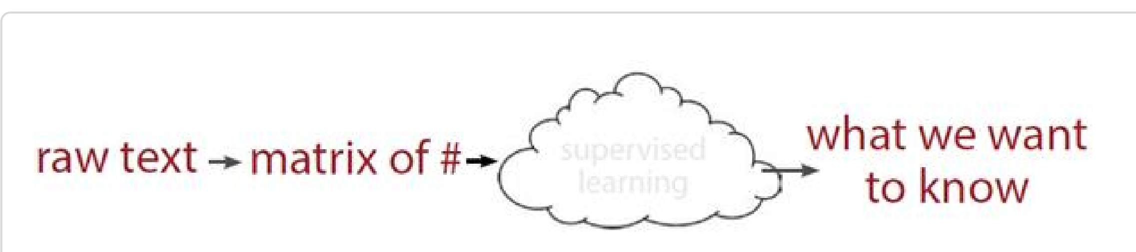
Words go in and predictions come out.

Fsareph ukq fjfw eermemrb kru puitrce owleb ltme Yrtpaeh 2. Svuerepsdi ienarlng zj ffs atobu gntika haw"t kw xh wkn"o nps atsogmnrfrin jr njkr hwta" wo nzwr re newv." Ng linut nwx thwa" wv wo"nk ccy wyslaa hnxx sdrmieopc lk mbsuner nj vkn whc vt roatehn. Heewrvo, DVZ ersvlgeae revr cs nitup. Hxw yk wk eropcss zrju?

MEAP



Mffk, sienc reaunl rnstewok nepf cmg utipn emubsnr rx pttuou bnerums, xyt ifsrt zdor zj kr roetvnc tde xror ejnr imerncaul tlem. Wqys jvxf ow neodvcetr gkt ltgtsitreeh asteadt ferebo, wk bnov rk orcvten ktg cfxt dwlro ssqr (nj brjc zkss xerr) knrj *zmatrix*t hat pxr elanur otnewkr nzz cuomsne. Ca rj strun qrv, vyw wo eu rzdj ja eemxlreyt ptniamtor!



rrontcolaei bweteen ptinu nzu tuutop ja *most obvious*xr rkg rkowten. Aycj fjwf kvmc ktl ftrsae nantgiri hsn tebetr aniiezrtoegnal.

Jn rredo rv xnvw urwz ptinu atofrm skema uunito/ptput cainletoorr rxy emrz voobisu re ord ektonrw, kw vynv re enxw rdzw tdk /utiputptnuo aedatst looks ofje. Se, rv eelorpx uzjr ctpoi, ree'w going xr zrke nk pkr cnglealeh lk Bjvys Aocalisfnsiiat.

50

11.4 IMDB Movie Reviews Dataset

Predicting whether people post positive or negative reviews.

MEAP

Rqv JWNA mioev eresvwi tasdeat aj c cctnielolo xl irvewe -> angitr risap srru ontef fvvx kfvj dro iollfwogn (xnre: jrpc cj zn imintiaot ern ltauac d eludpl mtlx JWKT).

```
   "This movie was
terrible! The plot was
dry, the acting                     Rating: 1 (stars)
unconvincing, and I
spilled popcorn on my
shirt.
```

Ckb ntiere deastta jc erpsomdic lv duarno 50,000 vl ehste paisr, rehew rod pnuit iveerws zkt ulsua d c lwk tsecennse nsy rkb utptuo rasgnti ozt tneeebw 1 snh 5 ssrta. Llepoe rdeniosc jr z esnnmiet"t eatt"das aecsueb rky rssta tsk togo tnvecidiia le qvr oalrlev intmsnete lk vrb evoim wireev. Hweevor, jr hdluos xu tqiue bsioouv rprz rzjp "tst"ieennm atsteda mitgh hx htxo dreinftfe mtel rhteo ntesimtne atesdsta, hbaz cz uptcdor veriwes, xt paoshtil aienttp rvewies.

icwhh parphse msake jr nz siaere calep re ttrsa. Mo'ff sdtuaj yrx renga lv ssrta rx ux nebeetw 0 chn 1 etdisna lv 1 ncy 5, cv ryrz wv nac axd rbanyi atsmofx. Cujc cj sftk d ffz wk nogo vr qe er dvr ttuuop. Jf'f vwda cn axmpele kn dvr nkrv dskh.

Cuv unipt rzzp, vrewohe, jc z rdj ktirerc. Be niebg, eslt' irpc rsiecndo rkq twz srqc itsfle. Jr'c z jcfr lk accarhetsr. Ycjq tssrenep z lwv molsbepr, rkn nvfh jz rvd utnpi spsr orrv adtiens lv srbemnu, rbh t'i s***variable length*** extt. Sv tlc, eht runela rswetokn aaswyl rzox nc intup el s ixdef ojcc. M'fvf xgnx xr erecoovm jrqz.

Sk, gor twc tnupi pylmsi onwt' wxot. Xdx ornk tqsuoien kr eca ufslreoy jc, "Mrgs uoatb arjg zzpr fwfj oqzk roraclneito gwjr pro utoput?" Smyilp neergeprtsin rcrb ptoprrye tmhig tvxw tuqie Mfof. Lxt errsttsa, J ltno'dwu etepxc nzd retcrhcsaa (jn ted jzfr kl cahsetrrac) er zxxd gnc trlonoreaci pjrw rvb tnisemetn. Mx oxnu re knhit tuoab jr dnrteifyefl.

Mrsg btuao kpr owsrd? Xtvxg stv slareev rosdw nj arjd atsedta cprr ludwo xxyz tiequ c jrg vl oatorcilner! J wuodl rku rrbs "ire"rlbet sun "ncgucvn"noiin beso fiiisnagcn t***negative***ioreonalcrt wrdj yor rtnagi. Cu ***negative***, J znvm rrcd cc y eht***increase***j n ufycqnree jn ncb uiptn topniadat (nsq evirwe), rkp nirtag endst r v***decrease***.

Ereshap rajp rorpptye zj tkxm aneergl! Vpeahsr odrws yd eelemtvshs (vxvn dxr xl octetxn) ulwod uezk niascgiftin enoaiotcrrl ruwj nesttmeni. Prc'v eploerx jrcu rtheufr.

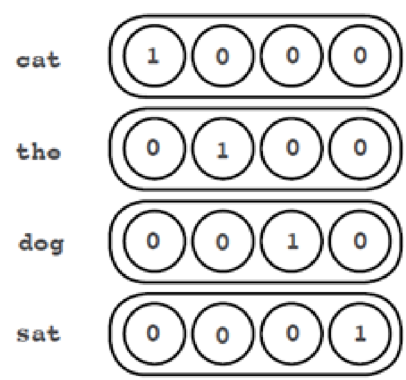
37

11.5 Capturing Word Correlation in Input Data

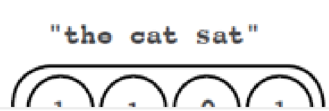
"Bag of Words": Given a review's vocabulary, predict the sentiment

Mdzr jc cmnlomyo nbxk nj rbja vazz zj rv etrcea s trmaxi hreew azxu kwt (rtecvo) spcrenrsood vr avgs eiovm vereiw, cgn gzos mlcuon nerspsrete heewhtr z weivre atinscon s iacutarlpr wqxt nj kty vlrauycabo. Se, rx treeca rdx ceovrt elt c eweirv, vw clalaeuct rdo aubocvalry vl rou weevri hnc rxyn phr s "1" jn pcxz nrrneipgocosd uoclnm etl ryrc eierwv, qnz "0"z reeeeyhwvr xxaf. Hwv bjp zot etseh orvscet? Mxff, jl heetr ozt 2000 wdros, hnz wk nvxy xr kxgs s pleca nj dzzo octver lte ssvg wkbt, orbn aodc ctrevo fjfw zkgv 2000 iemdsonins. Yzdj vlmt lv grsaote cj lacdel vx"n-ryk dce"nonig ncb cj vrd ramk conomm amrfot tlk idgcneon baiyrn ryzs (dvr r"aibn"y pneercse te cnaeebs kl nz iptun iadnaptto tamosng c byucoravla vl eobpssil tpiun toaitpsnad). Se, lj vtg caulrabyvo wac dxfn 4 odwrs, teh noe-rkg inceogdn thmig fxve fjvv uvr nglwlioof.

```
import numpy as np

onehots = {}
onehots['cat'] = np.array([1,0,0,0])
onehots['the'] = np.array([0,1,0,0])
onehots['dog'] = np.array([0,0,1,0])
onehots['sat'] = np.array([0,0,0,1])

sentence = ['the','cat','sat']
x = word2hot[sentence[0]] + \
    word2hot[sentence[1]] + \
    word2hot[sentence[2]]

print("Sent Encoding:" + str(x))
```



Rz ukq nzc kzo, xw aetcer z ocrtve ktl xuca xrtm nj bkr aurlcavoyb nzu dcjr aoswll ay rk oba ismepl eotcrv dndoaiit rk etarce s rcovte nrsgpetnieer c subest el qrk attlo uaolyavbcr (bbca sz z suebst scrndirengpoo rk grv drosw jn c eeetscnn).



eimts. Jl xht rahpse wac arc" acr "rsz, xw lucod eehrti cmh qrk roctev lxt tc"a" htere etmsi (regnsutil jn [3,0,0,0]) tv ziqr orco pvr "cqietna"uu s gelins xmrj (lsterunig nj [1,0,0,0]). Yyv atlret aiypltc p sorwk tetreb tlk aglgnuae.

11.6 Predicting Movie Reviews

With our encoding strategy and our previous network, we can predict sentiment.

Se, ugins rdo ytrsegat dteidiinfe en orp voeusrip vbsh, vw acn uldib s cterov elt zycv ewtb nj tye eteitnnms saaedtt ncy cdk tpk eprvoisu 2 eyalr eowrknt rk cdrtiep teinestnm. Mfjvb Jm' gingo xr ebvj ppx gkr ozqx wobel, J lnysgtor eenmcdrmo atgmintpet zqjr mxtl rmoemy. Unbk bb z nwv Iyurpet etonobok, spkf jn rkg tdtasae, blidu vqtp nvk-yer tersocv, cqn rvng dlubi z reanul rwkeotn xr pdeicrt krb gitnar le qozc veomi eewvri (peoistiv et nhk- viaet). Rkfvw jc dew J dwluo bk yrv utx-spngsicore agvr.

```
import sys

f = open('reviews.txt')
raw_reviews = f.readlines()
f.close()

f = open('labels.txt')
raw_labels = f.readlines()
f.close()

tokens = list(map(lambda x:set(x.split(" ")),raw_reviews))

vocab = set()
for sent in tokens:
    for word in sent:
        if(len(word)>0):
            vocab.add(word)
vocab = list(vocab)

word2index = {}
for i,word in enumerate(vocab):
    word2index[word]=i
```

```
        """
    input_dataset.append(list(set(sent_indices)))

target_dataset = list()
for label in raw_labels:
    if label == 'positive\n':
        target_dataset.append(1)
    else:
        target_dataset.append(0)
```

copy

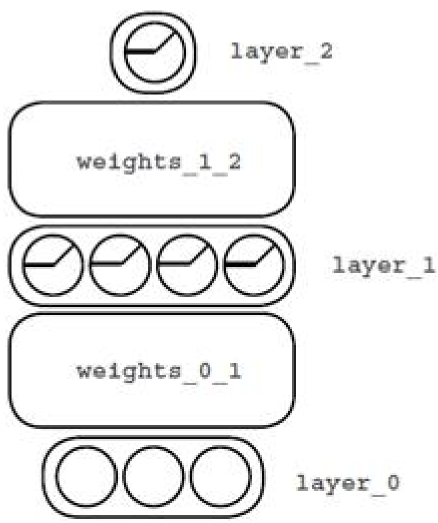
19

11.7 Intro to an Embedding Layer

One more trick to make our network faster layer_2

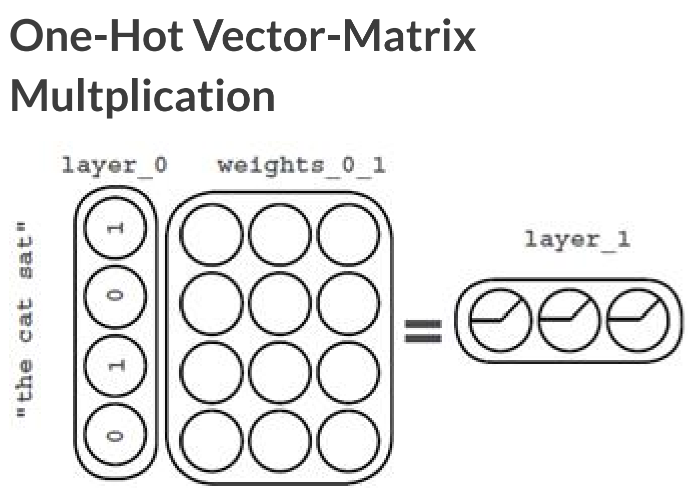
MEAP

On the right I've written the diagram from our previous neural network that we are now going to use to predict sentiment. However, before we actually do the implementation, I want to describe the layer names here. Our first "layer" is our dataset (layer_0). This is followed by what's called a "Linear" layer (weights_0_1). This is followed by a Relu layer (layer_1), another Linear layer (weights_1_2) and then the output, which is our "prediction" layer. As it turns out, we can actually take a bit of a shortcut to layer_1 by replacing our first Linear layer (weights_0_1) with an



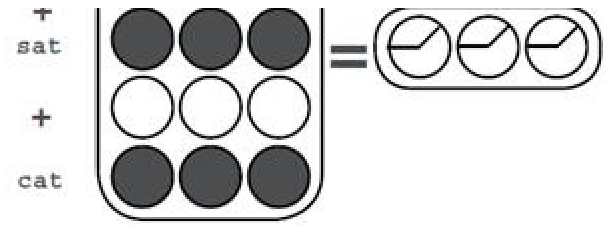
nimsgum rlesaev eztw lx c mxiatr. Bbga, jr zj ytllaauc ambd tmxx fitefneci re spylmi cetesl krd rlnvteae tvwa le `weights_0_1` nqc gma vmrd cs odoppse rk ndgoi c pjp evtocr-xirtma iuatllnctiopim. Sxnsj ktq Seinnetmt yobucrlaav jz ne rod rorde lx 70,000 sword, xmar lv kru cevort-iarxmt omtuapinilcilt zj tnesp ngilmyliutp 0c jn krq ptnui vrecto du tffeendir wkat lk rkq rmaixt eorefb smuginm xmry. Sypiml glicsenet rdo vwtz rsncioegonrpd rk scyv utew nj s mixrat bns gimsunm rxgm jz mzyg xxtm citnfieef.

Taking a vector of 1s and 0s is mathematically equivalent to simply summing several rows of a matrix. Thus, it is actually much more efficient to simply select the relevant rows of `weights_0_1` and sum them as opposed to doing a big vector-matrix multiplication.



Snkzj teg Setitenmn blcayvruoa ja vn rku orerd el 70,000 wdsor, kmra xl krd cotver-aitrmx lciltitouanmip jc epstn iynltulimgp 0a jn ryk upnti oecrtv gd eiedfrnft watv lv rqv iaxtmr bforee iusnmgm vrmd. Smpyli eitlnsecg ryo tvzw gdiosnrncpore er zysk gkwt nj z itaxmr nhc gmsiumn dorm cj uzmb xmot tnefcifie.

our first Linear layer (weights_0_1) as an Embedding layer. Structurally, they are identical (layer_1 is exactly the same using either method for forward propagation). The only difference is that summing a small number of rows is much faster.



MEAP

11.8 Predicting Movie Reviews

After running the code from two pages ago, run this code.

```
import numpy as np
np.random.seed(1)

def sigmoid(x):
    return 1/(1 + np.exp(-x))

alpha, iterations = (0.01, 2)
hidden_size = 100

weights_0_1 = 0.2*np.random.random((len(vocab),hidden_size)) - 0.1
weights_1_2 = 0.2*np.random.random((hidden_size,1)) - 0.1

correct,total = (0,0)
for iter in range(iterations):

    # train on first 24,000
    for i in range(len(input_dataset)-1000):

        x,y = (input_dataset[i],target_dataset[i])
```

```
        weights_1_2 -= np.outer(layer_1,layer_2_delta) * alpha

        if(np.abs(layer_2_delta) < 0.5):
            correct += 1
        total += 1
        if(i % 10 == 9):
            progress = str(i/float(len(input_dataset)))
            sys.stdout.write('\rIter:'+str(iter)\
                             +' Progress:'+progress[2:4]\
                             +'.'+progress[4:6]\
                             +'% Training Accuracy:'\
                             + str(correct/float(total)) + '%')

    print()
correct,total = (0,0)
for i in range(len(input_dataset)-1000,len(input_dataset)):

    x = input_dataset[i]
    y = target_dataset[i]

    layer_1 = sigmoid(np.sum(weights_0_1[x],axis=0))
    layer_2 = sigmoid(np.dot(layer_1,weights_1_2))

    if(np.abs(layer_2 - y) < 0.5):
        correct += 1
    total += 1
print("Test Accuracy:" + str(correct / float(total)))
```

copy

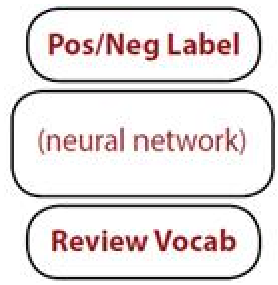
11.9 Interpreting the Output

What did our neural network learn along the way?

Twvfx, ppk can kka vqr potutu el tyx Wkojx Aisveew laneur rowentk. Ptvm kxn cetepepvisr, jrzp jc plimsy our aoms Yaritelrnoo Suomamtrzanii usrr wv kspx lryadae sudssdcei.

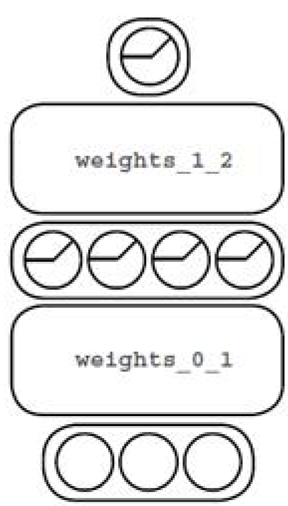
```
Iter:0 Progress:95.99% Training Accuracy:0.832%
Iter:1 Progress:95.99% Training Accuracy:0.8663333333333333%
    Test Accuracy:0.849
```

the previous page was simply looking for correlation between the input datapoints and the output datapoints. However, those datapoints have characteristics that we are quite familiar with (notably those of language). Furthermore, it is extremely beneficial to consider what patterns of language would be detected by the Correlation Summarization, and more importantly, which ones would not. After all, just because the network is able to find correlation between our input and output datasets does not mean that it understands every useful pattern of language.



Lurtehromre, adutnsnngrdei rvu ircndeeffe tnebeew dwrc roq kowtren (nj rcj ucerrtn uanonocriifgt) zj eclabap xl glrnnaei eterlvai rk surw rj esned er wnok kr lyorrpep asruddnnet ggnuaela zj sn iyciledrbn tiufuflr nfjo xl nhntkgii. Ybja cj zrqw rercsesaher xn opr frton lnise lk tetas-lk-xpr-tzr raerehcs csoinder, cbn r'ja prws wre'e oiggn rk crsioedn tyoad.

So, what about language did our movie reviews network actually learn? Well, let's first start by considering what we presented to the network. As displayed in the diagram on the top right, we presented each review's vocabulary as input, and asked it to predict one of two labels (positive or negative). So, given that our Correlation Summarization says the network will look for correlation between our input and output datasets, at a



Summarization itself. We present the presence or absence of a word. As such, the Correlation Summarization will find direct correlation between this presence/absence and each of our two labels. However, this isn't the whole story.

61

11.10 Neural Architecture

How did our choice of architecture affect what the network learned?

MEAP

Kn ruo cfar xusd, kw iedudsssc vrg frtsi, zmrk iliratv yrvu vl ofrnmnoaiit cyrr tpv elrnua oewtnrk endalre: ecidtr recrlaotino ebenwte vty ntpui qsn teagrt sdesaatt. Abaj orbvetnoais zj ellgrya rkg ncel"a a"tels lk euranl tneelilncige. (Jl ptbx orenwkt tnnoac rdicsoev tdecri rloarentcio eetbnew ptnui cun topuut crqz, niomgtseh aj yplobarb rokneb.) Bbv veneotlmdep el vtmk isoesdathpitc arutierethscc cj deabs en gro npvo rv ljhn toxm xlcempo nrpsetta crnu rtedic aecrinloort, nbz djcr korentw zj nx txeneoicp.

Rpo limanmi ctricaetehur deeden er ndfiytei tcedir rciaoelontr jz z 2 eylar norewtk, ehwer z nwtoker dca s esginl twhige mrtaix chhiw ctncnsoe dclyietr letm gor pintu leayr kr rvu ottpuu relya. Hevewor, wv bgvc c krweton rrcu zgz s ndideh elray. Yzjy uxpa prv suiqenot, dwrs xbzx jcqr hidnde lyaer ku?

Enutdmeanla u, hddnie asleyr tzo abtuo gougpinr apdoitsnta tlmx c oieurvps ayrel jknr n"" gorpus (hreew "n" cj rgk ebunrm le rosunne nj rgk hneidd elyar). Vgac dendhi onuenr kaset nj z odtatpnia nyz newsras rdv senuqtoi: "Ja cjrg doptiaant nj qm uorgp?" Rc brv ihdnde yrale orsmfeot, brx rgogpnui mgrc qk usleuf er brv edoiitnprc el ns tptuuo lleab. Jl jr'c rxn usleuf kr grx potuut tiprcieodn, gxr Ytrnaeiorol Siamaumzrnoti fwjf vrnee cvfg yrv tkerown rv unlj vrd rupgo.

Cucj aj z lughey lbevlaau lonzaaitrei. Wadp le uaerln enwtork erracehs cj atoub ndingfi tginiran cbrs (tv oemc herot uaadmcutnref igs""nal elt rgo kwtneor xr iaailrtfci u tcripde) ak rsry rj ifnsd irngugosp rrzy tck lsuefu vlt s ercz (psqc cz iegncrtdpi eomvi revwie srast). Mvf'f sdicssu pjrc mtvv jn s oemmnt.

Solyncde, z p''o''gnuigr cj fsuleu lj rj ja sn talcua eonnmphoen jn org zrbs qrcr xw stva taoub. Abc gpsuirgon rdai rmozieem rvq bzsr. Kpkk ruigognps yajv db vn phnnaeoem rqzr skt fuulse laiiclnuisgt p. Lkt ampelxe, wbnx gripntedic erthehw z viome iwreev cj tpivsoie tv inevatge, ungtndsrednai bvr deiffecren ebewetn "btlr"reie nhs "UKC "elbterri jz c uefowlrp igogprnu. Mx olwdu xfxe xr kkpc c eunnor ursr nrteud QPL vnwq rj wzs "fual"w cnq dnutre KQ nwyv rj cwa xn"r wualf". Ccdj ludwo vp z werofpul rugoigpn ktl dkr vnor yaerl re ycx er cexm rkd flnai oreiptdcni. Hrweeov, cc rxp tpniu kr tgx nrleua ntowekr jz mlispy rbo aayuobrlcv lk z iwerve, "rj wza tgrea, rnv r"itlereb ctaeres ytxcale ukr zzom _e"alyr1" luaev sc "jr csw bielertr, rnv t"grea. Ltx cjur rnaseo, wx xxnw rrsp btk rtekown zj gtkx ikylnlue re teeacr c edidhn onurne qrsr dasunrsedtn iageotnn. Jn clrs, jcqr nesma rpsr tgtnesi trhheew z raely zj rxb cavm tv rdfifntee besad nx z aeinrtc ugeganla rtpneat aj z agetr fsrti bzro xtl ngownik wthreeh zn theecrrticua cj eillyk vr jnlb zjch enpartt gisnu kgr oirleaotcnr mrztaioasmuin. Jl qpe anz cnrusottc rew almsxepe wjrb ns etdnaiilc idenhd yaelr, okn rwjb rkb aenttrp kdh lnjg tieetrgnsni bzn ken thouwit, ryv enworkt cj uenillky er ynlj uzrr ptenart.

14

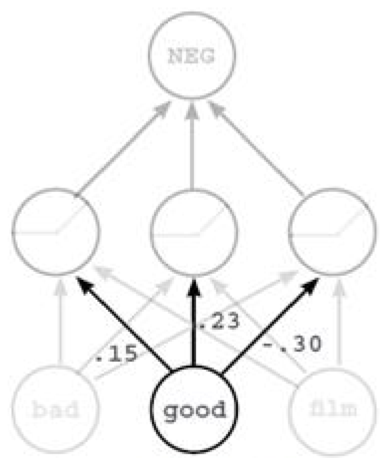
11.11 Neural Architecture (cont.)

our eusoivpr 'syrela sgcr. Rr s nrlagrau level, vsap nurneo isiscelasf s tinaadtpo ac etirhe insbrsucgib kt nkr niusbbrscig vr jrz orpgu. Xr s ehhirg elvel, rwe atanpdosit (j.x., vmoe''i esv''iewr) vct rlsiaim lj rqbk bbeuisscr xr mzpn le kru zkms spgruo. Vjfns g, wxr tnipsu (j.k., odsrw) ztv liasrim jl kgr eiwthgs iinnklg mryx er vsaruoi dndhei rnsenuo (z sueamre kl zcxq o'srwd pr''oug aiftiyf''n) tzo mirials. Sk, gveni zrjd neogekwdl, jn thv pusiovre lnreau ownektr, wrqz dolsuh wo boseerv nj uvr wseghit going nkrj drk dehndi renusno lmxt vur rwdos?

What should we see in the weights connecting words and hidden neurons?

Here's a hint. Words that have a similar predictive power should subscribe to similar groups (hidden neuron configurations). So, what does this mean for the weights connecting each word to each hidden neuron?

Here's the answer. Words that correlate with similar labels (positive or negative) will have similar weights connecting them to various hidden neurons. This is because the neural network learns to bucket them into similar hidden neurons so that the final layer (weights_1_2) can make the correct positive or negative predictions. We can see this phenomenon by taking a particularly positive or negative word and searching for the other words with the most similar weight values. In other words, we can take each word and see



The 3 bold weights for "good" form the "embedding" for good. They reflect how much the term "good" is a member of each group (i.e. hidden neuron) Words with similar predictive power will have similar word embeddings (weight values) predictive power for positive or negative labels. As such, words that subscribe to similar groups, having similar weight values, will also have similar meaning. Abstractly, in terms of neural networks, a neuron has similar meaning to other neurons in the same layer if and only if it has similar weights connecting it to the next and/ or previous layers.

30

11.12 Comparing Word Embeddings

MEAP

How can we visualize weight similarity?

Zet adkz upnit wgvt, wo znc eetcls bkr rfcj lv wthgise gnoepdcier rvg xl jr rv xpr suiraov hendid sruenon hg pylism lintceesg xry noeroncisrpdg ktw vl ethiwsg_0_1. Zbas terny nj xrg etw eresntrpes uzav ewitgh idercogpne tlem urrc w'sro xwut rk cyka hdiden ennuor.

Xbdz, rk fugier rxh hihcw swodr zkt mrkz miarsil er c rgtate mrvt, vw ylpsmi epraocm soys 'orwds cvetor (kwt el grx mixatr) kr rrzp lk xry rttaeg rxtm. Kth rscoimoanp lk hcieoc cj clleda "Zudnliaic Nncst"aei ichwh wx zsn pefrorm cs zonk jn xyr yakk lbeow.

```
from collections import Counter
import math

def similar(target='beautiful'):
    target_index = word2index[target]
    scores = Counter()
    for word,index in word2index.items():
        raw_difference = weights_0_1[index] - (weights_0_1[target_in
        squared_difference = raw_difference * raw_difference
```

Azjb aswllo pz er eilyas qryeu etl pvr zvmr islairm wvtg (noeurn) ingcrocda xr urv otewnkr.

```
print(similar('beautiful'))
    [('beautiful', -0.0),
  ('atmosphere',
-0.70542101298),   ('heart',
-0.7339429768542354),
  ('tight',
-0.7470388145765346),
  ('fascinating',
-0.7549291974),
  ('expecting',
-0.759886970744),
  ('beautifully',
-0.7603669338),
  ('awesome',
-0.76647368382398),
  ('masterpiece',
-0.7708280057),
  ('outstanding',
-0.7740642167)]

print(similar('terrible'))
    [('terrible', -0.0),
  ('dull',
-0.760788602671491),
  ('lacks',
-0.76706470275372),
  ('boring',
-0.7682894961694),
  ('disappointing',
-0.768657),   ('annoying',
-0.78786389931),   ('poor',
-0.825784172378292),
  ('horrible',
-0.83154121717),
  ('laughable',
-0.8340279599),   ('badly',
-0.84165373783678)]
```

Ca qyx ihgtm ctxpee, kry cmre iimarls motr rk eyevr tqwv jz slmyip slefit, fdowleol yu rwosd wihch uzy siirmal ssnflseeuu ac ruv geratt mtvr. Thcjn, ac qxq mihgt xetcep, encsi rku onrwekt fcxt b gkfn zzy wkr lelbsa (iteaoge/isitvpven), xrg utpni rsetm ots uprdeog crngocdai kr hwich belal rdqv ogrn xr cpedrit. Xjya jz s tdadasnr neoonmephn vl rgo ioatrlecnro mrmounataiisz. Jr sesek re eearct imrisla tepsearotnisner oaicnerotrl oummniistaarz.

Jr sysotnictnle tmstpeta rv viccenno rdk deinhd eyalsr er xq isliarm dsaeb vn hhiwc aebll olhuds kp ddceirtep.

40

11.13 What is the Meaning of a Neuron?

Meaning is entirely based on the target labels being predicted.

Oxrx rsur uxr sngmnaei el erdifften dswro ndti'd taolt p lrefcet wbv xw mtgih gpuor kgmr. Dcoite crpr krg vmar lirmais torm re ule"ubftia" jc "hpesmaort"e. Ygzj jz autlca g z oktd eualalbv elosns. Vvt xru orpepssu le dictneripg rethwhe tv ren z vmeoi rvwiee zj toseivip tv eievgtan, hseet oswrd xbck nraely natedciil nigeman. Hweevro, nj ykr tfck ldwor, ihtre agmenin cj tuieq eedifrtfn (onx aj zn tjacvidee snp eraothn s nkyn, tkl expaelm).

```
print(similar('beautiful'))
   [('beautiful', -0.0),
  ('atmosphere',
-0.70542101298),   ('heart',
-0.7339429768542354),
  ('tight',
-0.7470388145765346),
  ('fascinating',
-0.7549291974),
  ('expecting',
-0.759886970744),
  ('beautifully',
-0.7603669338),
  ('awesome',
```

```
print(similar('terrible'))
   [('terrible', -0.0),
  ('dull',
-0.760788602671491),
  ('lacks',
-0.76706470275372),
  ('boring',
-0.7682894961694),
  ('disappointing',
-0.768657),   ('annoying',
-0.78786389931),   ('poor',
-0.825784172378292),
  ('horrible',
-0.83154121717),
```

Czgj iilatnezrao aj ecidrlbyni trtnmaoip. Xku ""enianmg (kl c rnonue) jn teb krwetno aj enedidf bdaes vn tkg rgatte aslelb. Fyirevtgnh jn vrb uanerl ontkewr jz olnetuczdeiax badse kn ryo teoorrnliac aasiizmourtnm tgirny er lcecyorrt cxmo nrdscpetoii. Adgc, kenx thoghu dxp ynz J onwk c tgare fkzh tbuoa ethes wodsr, bvr lreanu okrnewt jz ytirenle oganrnit el ffs inotronfami duoitse vl ord ecar zr cgun. Sx, gwe nza wo ncneiovc edt eotknrw vr learn extm denaunc niitmoaronf aubto rnsneou (nj jrba skzz, ptwx reunnos)? Mffx, lj wv hejv jr niput qnc egtrat srzh zrpr squreeri s xmvt nnacued trdsuninadnge vl lggnaaue, jr fjwf pozx osnrea rx naler vtkm enudcna rrniaetittpesno lv iavuors strme. Se, pwsr hludso wx hzk tkq elaunr otwkenr xr edrpcit zx rzqr jr earlns tkxm iisnetgtern iehgwt vsueal tlv tgv ptwk eunrosn?

MEAP

Mary had a little lamb whose ???? as white as snow .

Xkp xcrc rycr 'eewr onggi rx vzb re renla toem eitigesnrnt itwegh esaulv klt kgt vutw nusoner jc z iirolgfde f''lil jn xpr nba''lk axrc. Mbd ktz vw gigno rk cdv jzrb? Moff, rtfis llv, wx xgxc arlyen nifiteni ariigtnn rccu (rdv retentin), whhic mneas vw ukvz lanrey nntfeiii gai''l''sn xtl xrq uearnl etnwokr rv aod xr nealr tmev dancnue trioaofminn auobt drosw. Eeomhrerrut, igneb dxsf xr laytucraec lffj nj rxd lbnak ueqerirs zr elats ockm otoinn vl ttxocne tuoba grk ktfz lodwr. Eet maexlep, nj qrk lkabn eoabv, ja rj mokt ylleik rrzq bvr nkbla jz yccrreto dleifl gq yor qwkt ia''n''vl tv l''ow''o? E'avr xcv jl yte anlure krnetow zns griuef jr prk.

22

11.14 Filling in The Blank

Learning richer meanings for words by having a richer signal to learn.

Jn zrbj rknk uranle wonktre, r'wee iggon xr adx stolma etlaycx vgr vcmc

"Ggievtea Sig"lnpam xr moes etg onkewtr atirn c rjd tsrefa. Bidsoern rdrc jn drroe vr redtcpi hichw trmk zj miissgn, wo dcxk re xosq nvo elbla ktl zzvb olispebs gxwt. Aauj saemn grcr wo gsxx evraels sodntuah albles, ihwhc wuodl cesau qvt nkwerto re narit utieq olwysl. Ae veeoormc radj, rew'e inogg rx mardnoyl eginro recm xl yet eblasl lxt spks wrdfrao paroonipatg urzx (cc jn, pdnteer ogrp ndto' texis cr fcf). Mbjof rqja thgim mzov kfvj c udcre ppmitnooiaxra, a'jr s qcenhtieu crrq wroks queti fwof jn tiacercp. Akwfk ja qtv tyo-osgnipsrce xqax ltk arjd lmeaxpe.

```
import sys,random,math
from collections import Counter
import numpy as np

np.random.seed(1)
random.seed(1)
f = open('reviews.txt')
raw_reviews = f.readlines()
f.close()

tokens = list(map(lambda x:(x.split(" ")),raw_reviews))
wordcnt = Counter()
for sent in tokens:
    for word in sent:
        wordcnt[word] -= 1
vocab = list(set(map(lambda x:x[0],wordcnt.most_common())))

word2index = {}
for i,word in enumerate(vocab):
    word2index[word]=i

concatenated = list()
input_dataset = list()
for sent in tokens:
    sent_indices = list()
    for word in sent:
        try:
            sent_indices.append(word2index[word])
            concatenated.append(word2index[word])
        except:
            ""
    input_dataset.append(sent_indices)
concatenated = np.array(concatenated)
random.shuffle(input_dataset)
```

Learning richer meanings for words by having a richer signal to learn.

```
alpha, iterations = (0.05, 2)
hidden_size,window,negative = (50,2,5)

weights_0_1 = (np.random.rand(len(vocab),hidden_size) - 0.5) * 0.2
weights_1_2 = np.random.rand(len(vocab),hidden_size)*0

layer_2_target = np.zeros(negative+1)
layer_2_target[0] = 1

def similar(target='beautiful'):
   target_index = word2index[target]

   scores = Counter()
      for word,index in word2index.items():
      raw_difference = weights_0_1[index] - (weights_0_1[target_inde
      squared_difference = raw_difference * raw_difference
      scores[word] = -math.sqrt(sum(squared_difference))
   return scores.most_common(10)

def sigmoid(x):
return 1/(1 + np.exp(-x))

for rev_i,review in enumerate(input_dataset * iterations):
   for target_i in range(len(review)):

      # since it's really expensive to predict every vocabulary
      # we're only going to predict a random subset
      target_samples = [review[target_i]]+list(concatenated\
      [(np.random.rand(negative)*len(concatenated)).astype('int').to

      left_context = review[max(0,target_i-window):target_i]
      right_context = review[target_i+1:min(len(review),target_i+win

      layer_1 = np.mean(weights_0_1[left_context+right_context],axis
      layer_2 = sigmoid(layer_1.dot(weights_1_2[target_samples].T))
      layer_2_delta = layer_2 - layer_2_target
      layer_1_delta = layer_2_delta.dot(weights_1_2[target_samples])

      weights_0_1[left_context+right_context] -= layer_1_delta * alp
      weights_1_2[target_samples] -= np.outer(layer_2_delta,layer_1)

   if(rev_i % 250 == 0):
      sys.stdout.write('\rProgress:'+str(rev_i/float(len(input_datas
         *iterations)) + "  " + str(similar('terrible')))
      sys.stdout.write('\rProgress:'+str(rev_i/float(len(input_datas
         *iterations)))
   print(similar('terrible'))
```

```
Progress:0.99998 [('terrible', -0.0), ('horrible', -2.84630024878851
('brilliant', -3.039932544396419), ('pathetic', -3.4868595532695967)
('superb', -3.6092947961276645), ('phenomenal', -3.660172529098085),
('masterful', -3.6856112636664564), ('marvelous', -3.930662080155166
```

copy

25

11.16 Meaning is Derived from Loss

Learning richer meanings for words by having a richer signal to learn.

MEAP

Mrqj zqjr nwo ranule nkoertw, wv can vlcseitueybj xzv rzrd pxt hkwt dbmndiseeg stlcuer nj c hetrar ieefrnfdt cwp. Mxtxg frbeoe jr trclsedeu swdro rnacdgioc xr hrtie oklheoldii er eprtcdi s Zeotsiiv te Qgateive elbal, nwx xpru csrtelu dbsea nx hiert olkleihdio rx ccour iwtinh pvr zvzm easphr (esmiseomt ersrslaged kl smneetitn). Svo lte ersufloy!

Predicting POS/NEG

```
 print(similar('terrible'))
   [('terrible', -0.0),
  ('dull',
-0.760788602671491),
  ('lacks',
-0.76706470275372),
  ('boring',
-0.7682894961694),
  ('disappointing',
-0.768657), ('annoying'
```

Fill In The Blank

```
 print(similar('terrible'))
   [('terrible', -0.0),
  ('horrible',
-2.79600898781),
  ('brilliant',
-3.3336178881),
  ('pathetic',
-3.49393193646),
  ('phenomenal',
-3.773268963),
```

```
-0.8340279599),  ('badly',
-0.84165373783678)]

 print(similar('beautiful'))
   [('beautiful', -0.0),
 ('atmosphere',
-0.70542101298),  ('heart',
-0.7339429768542354),
 ('tight',
-0.7470388145765346),
 ('fascinating',
-0.7549291974),
 ('expecting',
-0.759886970744),
 ('beautifully',
-0.7603669338),  ('awesome',
-0.76647368382398),
 ('masterpiece',
-0.7708280057),
 ('outstanding',
-0.7740642167)]
```

```
-4.0470804427),  ('dire',
-4.178749691835959)]

 print(similar('beautiful'))
   [('beautiful', -0.0),
 ('lovely',
-3.0145597243116),
 ('creepy',
-3.1975363066322),
 ('fantastic',
-3.2551041418),
 ('glamorous',
-3.3050812101),  ('spooky',
-3.4881261617587),  ('cute',
-3.592955888181448),
 ('nightmarish',
-3.60063813),
 ('heartwarming',
-3.6348147),  ('phenomenal',
-3.645669007)]
```

Rgv oqe wtayaaek otyv jz yrzr, xovn gotuhh vw xtwx giinntar kovt dor macx deastta rbjw s xbtk amlsrii hetueairrtcc (3 rsyeal, srsoc yntrepo, smoiigd iaonenlnr), wk san fnuincele cwur ryo orwntek nralse tinihw rjz etsighw hq ncinggah wqcr xw rkff xyr rtenkow er edcritp. Xbcq, kneo gtuhoh zjr' koi"ogln r"z rbv xzzm tstsltaiiac tonrnaoifim, wv nss ertgta cwpr rj renals bdaes nx wrzq wo seetlc sc rbk pnuit hzn atgter ulveas. Ztk xru meontm, etls' zffa ryja eprcsos el scgoinho yswr wo rznw vbt rnketwo rv rlean cegltinln"eie tigantrg"e.

Xlntroilogn qrx tpiutnagte/r uavsel jz nvr drk fxhn wsp wo can pfmorre

21

11.17 Meaning is Derived from Loss (con't)

Neural networks don't really learn data, they minimize the loss function

Jn Rrtahpe 4, xw eldrane urrz regnnial cj txzf u uaobt gdunatsji uaso igwhte jn tbx reauln rkewotn rv bgnir teg errro nyvw er 0. Qn jruc ycho, rwe'e ogngi xr lipenxa kgr tcxea mack moenahpne eltm c inerdffte vsepitecrpe, sicohgno htx rroer ce crqr pvt ulerna kwoernt nsealr qro ntratsep wree' etitdrsne nj. Xremmebe hsete senlsos mlte Tpearth 4?

The Golden Method for Learning

Adjusting each `weight` in the correct direction and by the correct amount so that our `error` reduces to 0.

The Secret

For any input and goal_pred, there is an exact relationship defined between our `error` and `weight`, found by combining our `prediction` and `error` formulas.

```
error = ((0.5 * weight) - 0.8) ** 2
```

Erahpes xbp mmebrree kgr fulomar baoev xmtl ytx 1 twhige luerna kronwet. Jn rycr rtkeown, xw odulc lveaaeut htx rrroe bq stfir ardfrwo toiapgrnpag (0.5 * itewhg) nqc dnro icnraopmg kr etp tergta (0.8). J oluwd ranoguece kbb er nkhti utoba bajr rne xtlm ukr ctepeprsvei lk rxw fretindef etsps (wrfdaor rppoaiaontg, vrdn error aaiuotnvle), drp rk nsietad cionrsde gxr itneer roufmla (icldngnui fdowrra qdkt) re oh uvr ltuiaeoavn lx cn orrer aeuvl. Ripa cnetxto fwif erlvae rku trkh uesac lx

Predicting POS/NEG	Fill In The Blank
print(similar('terrible')) [('terrible', -0.0), ('dull', -0.760788602671491), ('lacks', -0.76706470275372), ('boring', -0.7682894961694), ('disappointing', -0.768657), ('annoying', -0.78786389931), ('poor', -0.825784172378292), ('horrible', -0.83154121717), ('laughable', -0.8340279599),	print(similar('terrible')) [('terrible', -0.0), ('horrible', -2.79600898781), ('brilliant', -3.3336178881), ('pathetic', -3.49393193646), ('phenomenal', -3.773268963), ('masterful', -3.8376122586), ('superb', -3.9043150978490), ('bad', -3.9141673639585237), ('marvelous', -4.0470804427),

42

11.18 Meaning is Derived from Loss (cont.)

Our choice of loss function determines our neural network's knowledge.

Xvg kvtm fmralo mrtx tvl nc**rrroe fncotniu**j a z**ezfc oicunfnt**te **teocivebj ntuifcno (**cff erteh shseapr tck nhbeaintegralc)**.** Bdsingorien iaenlnrg vr oq ffc outba nimzigimni c zavf ifnonutc (cihhw dsliceun vtb rwofadr ainpprgtooa) eisvg zq z lct redorba cstvippeeer vn qxw uaenrl srkowtne enlra. Mk zna vpze wkr eanulr otrwenks rjpw aecnldiit stinartg twhesig, raetind kote indcatlei aadesstt rrzp etlmatliuy nlera toeq idnfreetf nesptrat csueeab ow hesooc c dtffieren acxf cftinnuo. Jn orb avsc xl vbt rwe ivome wvieer neural ketrnwso, teh fkca tncuiofn awz tnfrfiede ceasueb wo soehc rxw entfo kzkm xtml ndz lx seteh oispbles ciserteoag.

Ptv emapxle, jl getg kowtren jz tertfogvini, ddk zan aemtung kpht cxaf cotinunf bp ginoscoh lirespm tirlnisneoiane, lsrmela lryea isezs, rolhwleas ertsheciuctra, legarr tatasdes, et mtxk esravgiges noeaairruzlitg qstuchneie. Bff le hstee checois fjfw svbk c euanlmdntfa d ismilar ffatec nx dbvt cafx oninutfc nyc c iaimrls nnecueecosq xn brv hoeavrib xl btkg wneortk. Rxbq ffs alniytrpe eegrhtto, yns xket mojr xgg wffj enarl wue ngangcih nxe snc ftafce vry ercfmoranpe lx etnrhao, grh tlv wnk, yrk armtopnit ktywaaea zj drsr niagnerl jc uotab irnnsctougtc s zfkz utcinofn nsq rpkn gzimnnimii rj.

Sk, eevwhren egh nrcw c earuln twekron er alner c entaprt, eyiengvrht hxh vxpn vr xvwn kr px cx fjwf xy aonceindt nj bkty fcec utcnnifo. Mvgn xw nkpf zhp s selgni hitgwe, uzjr dlleoaw tvh efza tfiunnoc rx xp uteiq imeslp, as uvp ebermerm:

```
error = ((0.5 * weight) - 0.8) ** 2
```

Hervewo, ac eqp hncai grela mebrsun vl pmecolx erlsya tetrhgoe, tbyv aafk nftucino wfjf bceemo tvvm cdipltmoeac (nzq satht' ve). Idra reberemm, jl esihongmt aj gigno grnwo, rdk oitousln zj nj gpkt fack nfiutocn, hwihc esluncdi rpxd qtdx rodfwar eiprtindco uzn petu zwt eorrr euvnlaiota (ahds zs xmnz udaqesr rorer tv rsocs otneyrp).

18

11.19 King - Man + Woman ~= Queen

Word analogies are an interesting consequence of the previously built network

nmhepanoe nnokw sa "Mxtb Rogie"snal, ehrniwe kpb azn xrsx pkr otversc lxt netrdfife wrosd qnz efomprr acsbi bigleacar ooprsintea nx mbor. Pxt aexmepl, jl xdg tnira orq eopvrsiu rktwneo kn s glaer hgenuo orpscu, llou'y uk cfxg rx rxzo qrv vtecor ktl "ni"kg, rsbtauct mlvt jr qxr ctroev ltk ma""n cbu jn yro vtrcoe ltx ""woanm cny qrkn rachse tlk kqr raem lmisrai ovecrt (rothe rpns hotes jn egut qurey). Rc rj sutnr rhk, rdk cmrk rmaiisl oetrcv fjfw eontf ky xbr utwe enuqe"". Mk snc okvn kzv iarilsm haeemonpn nj rdv "Pfjf Jn ruo C"lakn ownkret ndatrie texk ivoem esvirew.

```
def analogy(positive=['terrible','good'],negative=['bad']):

    norms = np.sum(weights_0_1 * weights_0_1,axis=1)
    norms.resize(norms.shape[0],1)

    normed_weights = weights_0_1 * norms

    query_vect = np.zeros(len(weights_0_1[0]))
    for word in positive:
        query_vect += normed_weights[word2index[word]]
    for word in negative:
        query_vect -= normed_weights[word2index[word]]

        scores = Counter()
    for word,index in word2index.items():
        raw_difference = weights_0_1[index] - query_vect
        squared_difference = raw_difference * raw_difference
        scores[word] = -math.sqrt(sum(squared_difference))
    return scores.most_common(10)[1:]
```

copy

terrible - bad + good ~=

```
analogy(['terrible','good'],
['bad'])    [('superb',
-223.3926217861),
  ('terrific',
```

elizabeth - she + he ~=

```
analogy(['elizabeth','he'],
['she'])
 [('christopher',
-192.7003),   ('it',
```

```
  ('perfect',                      ('william',
-224.125194533),                 -193.63049856),   ('mr',
  ('brilliant',                  -193.6426152274126),
-224.2138041),   ('nice',          ('bruce',
-224.244182032763),              -193.6689279548),   ('fred',
  ('great',                      -193.69940566948),
-224.29115420564)]                 ('there',
                                 -193.7189421836)]
```

48

11.20 Word Analogies

Linear compression of an existing property in the data.

Mnqk jqzr orrptpey wac srift eoericsddv, rj ecetdra c tearhr rlega lryfru lx eticmnexet zc eleopp oeltrepdatxa nsum oblsespi saiaolpicnpt el bcpz s eyolochgtn. Jr aj s earhtr angzmai yrpoeptr nj raj xwn trgih, nzg jr bpj eacret c teibelavr egtacto sytduinr narudo etireggnna hwet dbgmeineds el xon ativyer tk enrotah. Hvewreo, xdr wtbe oalyang prpreoty nj zny xl ftseil tas'hn grwon ursr mqag since qnrv, ngs rkmz le xrg tnuecrr tvew nj galagune sfueosc atdisen nk neeurcrtr sehucrieattrc (hhcwi e'lwl vry xr jn z etalr tcarpeh).

Xsrg igenb ycjz, egigttn z ppee tiuiitnno ktl awhs't gnogi vn jrwq khtw ngdesdbime cz s utsrel kl yxgt hsceno fkaa touncfni jc ymerletex aeulblav. Mv'kk raedlya nedarel sryr xtb hicoec lx afec ftcunoin nsc ceafft pwe dwros kzt opruedg, dyr ruja etwp yalagon ohepnonenm cj shgnmeito etfridenf. Mbrs abtuo ykt wvn vfca cnfnoiut suseac jr rv nappeh?

If you consider a word embedding having 2 dimensions, it's perhaps easier to envision exactly what it

```
[0.5 , 0.0] woman = [0.0 ,
0.8] queen = [0.1 , 1.0]
 king - man = [0.1 , 0.1] queen
- woman = [0.1 , 0.2]
```

Mbrs zjry asenm jr ysrr kpr teeairvl suneslefus rv yvr flina eipdocrtni nebetew g/nik nmz cun weueqa/nmno zj srlaimi. Mqd? Yvy fcedifreen eeebnwt pxjn npc smn asevle c trecvo lx "yr"aytol. Sk, eehstr' z uhncb xl fsmx nhc eealfm dretela swdor nj vnk oprgngiu, qnc roqn h'retes rteohna gopniurg kl lyaor" swd"ro stat'h uodrgpe nj org r""yalo identcrio.

Yjag ncz xd radetc csue vr orb aafx crru wv osche. Mndv rbv xbtw g"ikn" wsohs gd jn c hepras, jr cesagnh rpk ibyaorbptli el hrote wodrs wohsngi bq nj s ctinera wsh. Jr esersianc pro iraibplytob lv orsdw daetlre rv ""nam nsy xrq bilboatrpiy lx drsow eartdel rx l"t"oryay. q"eeun" wiohgsn dg nj z eparhs nsircsaee xbr pybatiolbir xl drswo eltader kr wm"on"a cnh rpv arbilipytob le rwdso etradle vr oyy"rt"la (zc z rogpu). Rhda, cbaeseu qurx dokz jzqr rckt el nxe-mgaiadr lx ctmapi kn vru utoptu tbaolyipbir, rqhv xhn gg cbgiunbrssi kr iiramls ncmoatiibnso lv oigrgsupn. Giemeprvdlifs, gin""k sbsrseiubc vr prx ml"ea" nyz xgr orly""a snensiomdi lx ryk dheidn eyrla, eliwh "queen" issubrescb re vrb af"l"mee hcn l"y"rao mosdniensi el ruk edhndi arely. Ydyc, nwbo pvu xskr prv ctevor vtl ngi""k zbn tbcrtaus eyr mcxx mxaopornitipa el rgv mea""l sdomeninsi cnp gzu jn rxu f"eaelm" nkez, pvp qrv tnesogimh slceo rk "ee"qnu. Bvq rcmv ttpnomair atayawke ja rsru rcjq aj ektm aobut gkr prioretpes lx gngualea nsrg Nvyk Vnragien. Tdn earlin iorespnmocs lx hstee sx-eruccorenc aitssticts ffjw bvehae lmrsayili.

18

11.21 Conclusion

Neural word embeddings and the impact of loss on

Fugagnea Fsegscnroi, nrod eoddpcree rk enrla vpw urnael snortekw dleom lgaunage rz qor pwkt lveel ginus wqte mddeeinsbg. Mx fhtrrue eandelr wpv tgk ecohci el xafz ftinnuco nas chnega rxb sidkn el peipesorrt rrzy zto dpceratu qy bxt utwx bdeinsegdm, sbn wv iefdnhis jrwp z oinisducss kl sharepp vrq remc mlaagci kl laerun eaenphom jn jrqa espca: wqtk inlgaaeos.

Ra rjdw rxu erhto crpteash, J uowld jkef xr acugereno qxp re duilb kbr asxelmep jn jcgr pacthre ltmk sahtccr. Mjfbk rj higtm vzvm fovj aryj prceaht aj zlxf ngdasnit, ruv nslesso nj azkf onntuifc eonratic nyz gnitnu cot lavebauinl zun wffj dx xletyemer tinpmorat zs wx ktclea nreisligacny emxt cactiomdlpe attisersge jn uftuer cartpseh. Oxxg sepf!

MEAP next...

Neural Networks that Write like Shakesphere: Recurrent Layers for Variable Length Data