transformers natural build train fine tune deep neural network architecture nlp python pytorch tensorflow bert gpt-3 transformers natural language processing right reserve book reproduce store retrieval system transmit form mean prior write permission publisher case brief quotation embed critical article review effort preparation book ensure accuracy information present information contain book sell warranty express imply author packt publishing dealer distributor hold liable damage cause allege cause directly indirectly book mention book appropriate use capital packt publishing guarantee accuracy information producer tushar gupta acquisition editor peer review saby dsilva project editor janice gonsalves content development editor bhavesh amin copy editor safis editing technical editor karan sonawane proofreader safis editing indexer pratik shirodkar presentation designer pranit padwal publish january 2021 second edition march 2022 production reference 3170322 publish packt publishing ltd.

35 livery street b3 2pb uk year transformers take nlp community storm break record achieve previous 30 year model bert t5 gpt constitute fun- damental building brick new application computer vision speech recognition translation protein sequencing write code reason stanford recently introduce term foundation model define set large language model base giant pre trained transformer progress thank simple idea book reference interested understand transformer work theoretical practical perspective author tremendous job explain use transformer step step hand approach read book ready use state art set technique empower deep learning application particular book give solid background architecture transformer cover detail popular model bert roberta t5 gpt-3 explain use case text summarization image labeling question answering sentiment analysis fake news analysis transformer cover topic interest definitely worthwhile book edition place desk go happen second edition engineering director office cto google author denis rothman graduate sorbonne university paris diderot university design patent encoding embedding system author patent ai cognitive robot bot begin career deliver natural language processing nlp

chatbot moët et chandon ai tactical defense optimizer airbus aero- advanced planning scheduling aps solution worldwide want thank corporation trust start deliver artificial intelligence solution share risk continuous innovation want thank family believe reviewer george mihaila ph.d. candidate university north texas department computer science get master degree computer science receive bachelor degree electrical engineering home country romania work 10 month tcf bank helped machine learning op- eration framework automatic model deployment monitoring internship state farm data scientist machine learning engineer work data scientist machine learning engineer university north texas high performance computing center 2 year work research field natural language processing 5 year 3 year spend work transformer model research interest dialogue generation persona technical reviewer edition transformers natural language process- ing denis rothman currently work doctoral thesis casual dialog generation persona free time george

work doctoral thesis casual dialog generation persona free time george like share knowledge state art language model tutorial article help researcher field nlp join book discord space join book discord workspace monthly ask session author table contents chapter 1 transformer ecosystem transformer 3 industry 4.0 3 foundation model 4 programming sub domain nlp 6 future artificial intelligence specialist 8 optimize nlp model transformer 9 background transformer 10 resource use 11 rise transformer 4.0 seamless api 12 choose ready use api drive library 14

choose transformer model 16 role industry 4.0 artificial intelligence specialist 16 summary 18 question 19 reference 19 chapter 2 getting start architecture transformer model 21 rise transformer attention need 22 encoder stack 24 table contents input embedding 26 positional encoding 29 sublayer 1 multi head attention 36 sublayer 2 feedforward network 53 decoder stack 54 output embedding position encoding 55 attention layer 55 ffn sublayer post ln linear layer 56 training performance 57 tranformer model hugging face 57 summary

match tokenizer dataset 503 chapter 10 semantic role labeling bert base transformer 504 chapter 11 let data talking story question answer 505 chapter 12 detect customer emotion prediction 506 chapter 13 analyze fake news transformer 507 chapter 14 interpret black box transformer model 508 chapter 15 nlp task agnostic transformer model 509 chapter 16 emergence transformer drive copilot 510 book enjoy transformer game changer

natural language understanding nlu subset nat- ural language processing nlp pillar artificial intelligence global digital economy transformer model mark beginning new era artificial intelligence language under- standing pillar language modeling chatbot personal assistant question answering text summarizing speech text sentiment analysis machine translation witness expansion social network versus physical encounter e commerce versus physical shopping digital newspaper streaming versus physical theater remote doc- tor consultation versus physical visit remote work instead site task similar trend hundred domain incredibly difficult society use web browser stream service digital activity involve language ai language understanding paradigm shift society physical massive digital information force artificial intelligence new era artificial intelligence evolve billion parameter model face challenge trillion word dataset transformer architecture revolutionary disruptive break past leave dominance rnn cnn bert gpt model abandon recurrent network layer replace self attention transformer model outperform rnns cnns 2020s experience major change ai transformer encoder decoder contain attention head train separately parallelize cutting edge hardware attention head run separate gpu open door billion pa- rameter model soon come

trillion parameter model openai train 175 billion pa- rameter gpt-3 transformer model supercomputer 10,000 gpu 285,000 cpu core increase datum require train ai model scale transformer pave way new era parameter drive ai learn understand hundred million word fit sentence require tremendous parameter transformer model google bert openai gpt-3 take emergence level transformer perform hundred nlp task train transformer learn image

classification reconstruction embed image sequence word book introduce cutting edge computer vision transformer vision transformers vit clip dall e. foundation model fully train transformer model carry hundred task fine tuning foundation model scale offer tool need massive think human control content billion message post social network day decide legal ethical extract information contain think human require translate million page publish day web imagine people manually control million message minute finally think human write transcript vast hour streaming publish day web finally think human resource require replace ai image captioning billion image continuously appear online book develop code prompt design new programming skill control behavior transformer model chapter key aspect language understanding scratch python pytorch tensorflow learn architecture original transformer google bert openai gpt-3 t5 model fine tune transformer train model scratch learn use powerful api facebook google microsoft big tech corporation share large dataset explore close market demand language understanding field medium social medium research paper example hundred ai task need summarize vast amount datum research translate document area economy scan social medium post ethical legal reason book work hand python pytorch tensorflow introduce key ai language understand neural network model learn explore implement transformer learn new skill require industry 4.0 ai specialist disruptive ai era book aim reader knowledge tool python deep learning need effectively develop key aspect language understanding book book introduction python programming machine learning concept instead focus deep learning machine translation speech text text speech

instead focus deep learning machine translation speech text text speech language modeling question answering nlp domain reader benefit book deep learning nlp practitioner familiar python programming data analyst datum scientist want introduction ai language understanding process increase amount language drive function book cover introduction transformer architectures chapter 1 transformer explain high level transformer look transformer ecosystem property foundation model chapter highlight platform available

evolution industry 4.0 ai specialist chapter 2 getting start architecture transformer model go background nlp understand rnn lstm cnn deep learning architecture evolve transformer architecture open new era transformer architec- ture unique attention need approach invent google research google brain author describe theory transformer hand dirty python multi attention head sub layer work end chapter understand original architecture transformer ready explore multiple variant usage transformer follow chapter chapter 3 fine tune bert models build architecture original transformer bidirec- tional encoder representations transformers bert show new way perceive world nlp instead analyze past sequence predict future sequence bert attend sequence key innovation bert architecture fine tune bert model go step google colaboratory notebook like human bert learn task perform new one have learn topic scratch chapter 4 pretraine roberta model scratch build roberta transformer model scratch hugging face pytorch module transformer bert like distilbert like train tokenizer scratch customize dataset train transformer run downstream mask language modeling task ii apply transformers natural language understanding generation chapter 5 downstream nlp task transformer reveal magic transformer model downstream nlp task pretrained transformer model fine tune solve range nlp task boolq cb multirc rte wic dominate glue superglue leaderboard evaluation process transformer task dataset metric run downstream task hugging face pipeline chapter 6 machine translation transformer define machine translation understand human baseline machine transduction method preprocess wmt french english dataset european parliament machine translation require precise evaluation method chapter explore bleu

parliament machine translation require precise evaluation method chapter explore bleu scoring method finally implement transformer machine translation model trax chapter 7 rise suprahuman transformer gpt-3 engine explore aspect openai gpt-2 gpt-3 transformer examine architecture openai gpt model explain different gpt-3 engine run gpt-2 345 m parameter model interact generate text gpt-3 playground action code gpt-3 model nlp task

compare result gpt-2 chapter 8 apply transformer legal financial document ai text summarization go concept architecture t5 transformer model initialize t5 model hugging face summarize document task t5 model summarize document include sample bill rights explore success limitation transfer learning approach apply transformer finally use gpt-3 summarize corporation law text second grader chapter 9 match tokenizer dataset analyze limit tokenizer look method apply improve datum encoding process quality build python program investigate word omit misinterpret word2vector tokenizer follow find limit pretrained tokenizer tokenizer agonistic method improve t5 summary apply idea room leave improve methodology tokenization process finally test limit gpt-3 language understanding chapter 10 semantic role labeling bert base transformers explore transformer learn understand text content semantic role labeling srl challenge exercise human transformer produce surprising result implement bert base transformer model design allen institute ai google colab notebook use online resource visualize srl output finally question scope srl understand reason limitation iii advanced language understanding techniques chapter 11 let data talking story question answer show transformer learn reason transformer able understand text story dis- play reasoning skill question answering enhance add ner srl process build blueprint question generator train transformer stand solution chapter 12 detect customer emotion prediction show transformer im- prove sentiment analysis analyze complex sentence stanford sentiment treebank challenge transformer model understand structure sequence logical form use transformer prediction trigger different action depend sentiment analysis output chapter finish edge case gpt-3 chapter 13 analyze fake news transformer delve hot

case gpt-3 chapter 13 analyze fake news transformer delve hot topic fake news transformer help understand different perspective online content day day billion message post article publish web social medium website form real time communication available technique previous chapter analyze debate climate change gun control tweet president moral ethical problem determine consider fake news reasonable doubt news remain subjective chapter 14

interpret black box transformer models lift lid black box trans- model visualize activity use bertviz visualize attention head language interpretability tool lit carry principal component analysis pca finally use lime visualize transformer dictionary learning chapter 15 nlp task agnostic transformer models delve advanced model re- deberta run example hugging face transformer process image sequence word look different vision transformer vit clip dall e. test computer vision task include generate computer image chapter 16 emergence transformer drive copilot explore maturity industry 4.0 chapter begin prompt engineering example informal casual english use github copilot openai codex create code line instruction vision transformer help nlp transformer visualize world create transformer base recommendation system digital human metaverse end appendix terminology transformer models examine high level structure transformer stack sublayer attention head appendix ii hardware constraints transformer models look cpu gpu performance run- ning transformer transformer gpu transformer perfect match conclude test google colab cpu google colab free gpu google colab pro gpu appendix iii generic text completion gpt-2 provide detailed explanation generic text completion gpt-2 chapter 7 rise suprahuman transformers gpt-3 engines appendix iv custom text completion gpt-2 supplement chapter 7 rise suprahuman transformers gpt-3 engines build train gpt-2 model make interact custom text appendix v answer question provide answer question end chapter book program book colaboratory notebook need free google gmail account able run notebook google colaboratory free vm need python instal machine educational program necessary time read chapter 2 getting start

machine educational program necessary time read chapter 2 getting start architecture transformer model appendix terminology transformer models chapter 2 contain description original transformer build building block explain appendix terminology transformer models implement book find difficult pick general intuitive idea chapter chapter feel comfortable transformer chapter read chapter consider implement transformer customer use career novel idea download example code file code bundle book host github

https://github.com/denis2054/ transformers nlp-2nd edition code bundle rich catalog book video available https://github.com/packtpublishing/. check download color image provide pdf file contain color image screenshot diagram book download https://static.packt-cdn.com/downloads/9781803247335 text convention book codeintext indicate sentence word run model book code word text database table name folder name filename file extension pathname dummy url user input twitter handle example wish explore code find google colaboratory positional_encoding.ipynb notebook text.txt file chapter github repository block code set follow import numpy np scipy.special import softmax wish draw attention particular code block relevant line item set bold black cat sit couch brown dog sleep rug note use openai codex later book currently waiting list sign avoid long wait time https://openai.com/blog/ command line input output write follow 0.9627094

final positional encoding similarity bold indicate new term important word word screen example menu dialog box appear text like example case look t5 large t5 large model smoothly run google colaboratory touch feedback reader welcome general feedback email feedback@packtpub.com mention book title subject message question aspect book email questions@ errata take care ensure accuracy content mistake happen find mistake book grateful report visit http://www.packtpub.com/submit-errata select book click errata submission form link enter detail piracy come illegal copy work form internet grateful provide location address website contact interested author topic expertise interested write contribute book visit http://authors.packtpub warning important note appear like tip trick appear like share thought read transformers natural language processing second edition love hear thought click straight amazon review page book share feedback review important tech community help sure deliv- ere excellent quality content transformers transformer industrialize homogenized post deep learning model design parallel computing supercomputer homogenization transformer model carry wide range task fine tuning transformer perform self supervise learning billion record raw unlabeled datum billion

parameter particular architecture post deep learning call foundation model foundation model transformer represent epitome fourth industrial revolution begin 2015 machine machine automation connect artificial in- telligence general specifically natural language processing nlp industry 4.0 i4.0 go far software practice past year ai effective cloud service seamless api paradigm download library develop educational exercise industry 4.0

project manager openai cloud platform sign obtain api key work minute user enter text specify nlp task obtain response send gpt-3 transformer engine finally user gpt-3 codex create application knowledge programming prompt engineering new skill emerge model gpt-3 model fit specific task example project manager consultant developer want use system provide google ai amazon web services aws allen institute ai hugging face transformer project manager choose work locally implementation directly google cloud microsoft azure aws development team select hugging face google trax openai allennlp artificial intelligence specialist data scientist use api practically ai development answer know future employer customer user want specify ready adapt need come book describe offer exist market book provide reader solution adapt industry 4.0 ai drive nlp challenge chapter explain transformer high level chapter explain importance acquire flexible understanding type method implement transform- er definition platform framework library language blur number api automation available market finally chapter introduce role industry 4.0 ai specialist advance embed- need address critical notion start journey explore variety transformer model implementation describe book chapter cover following topic emergence fourth industrial revolution industry 4.0 paradigm change foundation model introduce prompt engineering new skill background transformer challenge implement transformer game change transformer model api difficulty choose transformer library difficulty choose transformer model new role industry 4.0 artificial intelligence specialist step explore ecosystem transformer ecosystem transformer transformer model represent paradigm change require new describe foundation model accordingly stanford university create center research foundation models crfm august 2021

crfm publish page paper reference section write scientist professional op- portunitie risks foundation models foundation model create academia big tech industry example goo- gle invent transformer model lead google bert microsoft enter partnership openai produce gpt-3 big tech find well model face exponential increase

petabyte datum flow data center transformer bear necessity let industry 4.0 consideration understand need industrialize artificial intelligence model agricultural revolution lead industrial revolution introduce machinery second industrial revolution give birth electricity telephone airplane industrial revolution digital fourth industrial revolution industry 4.0 give birth unlimited number ma- chine machine connection bot robot connect device autonomous car smartphone bot collect datum social media storage transformer turn million machine bot generate billion datum record day image sound word event show figure 1.1 figure 1.1 scope industry 4.0 industry 4.0 require intelligent algorithm process datum decision hu- man intervention large scale face unseen datum history humanity big tech need find single ai model perform variety task require separate algorithm past transformer distinct feature high level homogenization mind blow emer- gence property homogenization make possible use model perform wide variety task ability emerge train billion parameter model supercomputer paradigm change make foundation model post deep learning ecosystem show figure 1.2 scope i4.0 ai specialist foundation model design innovative architecture build history ai result artificial intelligence specialist range skill stretch present ecosystem transformer model unlike evolution artificial intelligence sum property model industrial layer model identical specifically design parallel processing architecture transformer chapter 2 get start architecture transformer model big tech possess huge data source history humanity generate industrial revolution digital boost unfathomable size industry 4.0 transformers big tech possess computer power see scale example gpt-3 train 50 petaflop second google domain specific super- computer exceed 80 petaflop second highly train transformer trigger task prompt prompt enter natural language word require structure make prompt metalanguage

foundation model transformer model train supercomputer billion record datum billion parameter model perform wide range task fine tuning scale foundation model unique fully train model call engine gpt-3 google bert handful transformer engine qualify foundation model let explore example foundation model work

engine qualify foundation model let explore example foundation model work change way programming sub domain nlp chen et al 2021 publish bombshell paper august 2021 codex gpt-3 model convert natural language source code codex train 54 million public github software repository codex produce interesting natural language source code chapter 16 emergence transformer drive copilot programming translation task natural language source code language programming nlp task gpt-3 engine let look example answer question bear mind codex stochastic algorithm metalanguage tricky generate expect careful engineer prompt correctly refer foundation model book mention openai gpt-3 google bert model gpt-3 google bert fully train supercomputer interesting effective limited use model reach homogenization level foundation model lack resource create prompt experiment codex example idea codex work purely educational purpose prompt generate random distribution 200 integer 1 100 python plot datum matplotlib create k mean clustering model 3 centroid fit model print cluster label plot cluster plot cluster centroid codex translate natural metalanguage prompt python automatically codex stochastic model reproduce exactly code try learn metalanguage experimentation drive like race car python program generate automatically copy test

import numpy np import matplotlib.pyplot plt sklearn.cluster import kmeans sklearn.datasets.samples_generator import make_blob generate random datum x y = make_blobs(n_samples=200 centers=3 n_features=2 cluster_std=2 plot datum plt.scatter(x 0 x 1 s=50 create k means model kmeans = kmeans(n_clusters=3 random_state=0 fit model datum transformer print cluster label plot cluster plt.scatter(x 0 x 1 c = kmeans.labels cmap='rainbow

plot cluster centroid plt.scatter(x 0 x 1 c = kmeans.labels cmap='rainbow plt.scatter(kmeans.cluster_centers 0 kmeans.cluster_centers 1 c='black s=100 alpha=0.5 copy paste program work try javascript exper- github copilot available microsoft develop tool chapter 16 emergence transformer drive copilots learn prompt engineering metalanguage reduce development time year come end user create prototype small task master metalanguage future code copilot expand codex fit future artificial intelligence chapter 16 point let glimpse bright future artificial intelligence specialist future artificial intelligence specialist societal impact foundation model underestimate prompt engineering skill require artificial intelligence specialist future ai specialist limit transformer ai data science overlap i4.0 ai specialist involve machine machine algorithm classical ai iot edge computing ai specialist design develop fascinating connection be- tween bot robot server type connect device classical algorithm book limit prompt engineering wide range design skill require industry

4.0 artificial intelligence specialist i4.0 ai specialist prompt engineering subset design skill ai specialist develop book refer future ai specialist industry 4.0 artificial intelligence specialist let general view transformer optimize nlp model optimize nlp model transformer recurrent neural networks rnns include lstms apply neural network nlp sequence model decade recurrent functionality reach limit face long sequence large number parameter state art transformer model prevail section go brief background nlp lead transformer describe detail chapter 2 get start architecture transformer model let intuitive look attention head transformer replace rnn layer nlp neural network core concept transformer sum loosely mix token nlp model convert word sequence token rnn analyze token recurrent function transformer analyze token sequence relate token token sequence show figure 1.3 figure 1.3 attention head layer transformer detail attention head chapter 2 moment takeaway figure 1.3 word token sequence relate word sequence model open door industry 4.0 nlp let briefly background transformer transformers background transformer past 100 + year great mind work sequence pattern language modeling result machine progressively learn predict

probable sequence word book cite giant happen section share favorite researcher lay ground arrival transformer early 20th century andrey markov introduce concept random value create theory stochastic process know ai markov decision process mdp markov chains markov processes early 20 th century markov show predict element chain sequence past element chain apply method dataset contain thousand letter past sequence predict follow letter sentence bear mind computer prove theory use today artificial intelligence 1948 claude shannon mathematical theory communication publish claude shannon lay ground communication model base source encoder transmitter receiver semantic decoder create information theory know today 1950 alan turing publish seminal article

computing machinery intelligence alan turing base article machine intelligence successful turing machine de- crypte german message world war ii message consist sequence word english rule system rule system program run list rule analyze language structure rule system exist case ma- chine intelligence replace rule list billion language combination automatically learn pattern expression artificial intelligence john mccarthy 1956 establish machine learn 1982 john hopfield introduce rnn know hopfield network associative neural network john hopfield inspire w.a. little write existence persistent state brain 1974 lay theoretical ground learn process decade rnn evolve lstms emerge know today rnn memorize persistent state sequence efficiently show figure 1.4 figure 1.4 rnn process state sn capture information sn-1 network end reach function f perform action transduction modeling type sequence base task 1980s yann lecun design multipurpose convolutional neural network cnn apply cnns text sequence apply sequence transduction modeling base w.a.

little persistent state process information layer layer 1990 sum year work yann lecun produce lenet-5 lead cnn model know today cnn efficient architecture face problem deal long term dependency lengthy complex sequence mention great name paper model humble ai spe- cialist everybody ai right track year markov fields rnns cnns evolve multiple model notion

attention appear peek token sequence add rnn cnn model ai model need analyze long sequence require increase computer power ai developer powerful machine find way optimize gradient research sequence sequence model meet expectation progress thirty year pass way start late 2017 industrialized state art transformer come attention head sublayer rnn appear pre requisite sequence modeling anymore dive original transformer architecture chapter 2 getting start architecture transformer model let start high level examine paradigm change software resource use learn implement transformer model resource use industry 4.0 ai blur line cloud platform framework library language model transformer new range number ecosystem mind blow google cloud provide ready use transformer model transformers openai deploy transformer api require practically programming hugging face provide cloud library service list endless chapter high level analysis transformer ecosystem implement book choice resource implement transformer nlp critical question survival project imagine real life interview presentation imagine talk future employer employer team customer begin presentation excellent powerpoint hugging face example adverse reaction manager sorry use google trax type project hugging face implement google trax game problem arise specialize google trax instead reaction manager want use openai gpt-3 engine api devel- opment specialize openai gpt-3 engine api development face project manager customer prefer hugging face automl api bad thing happen manager accept solution end work nlp task project section challenge face let begin rise transformer 4.0 seamless api industrialization era artificial intelligence microsoft google am- developer hope outperform tech giant million dollar supercomputer massive dataset train transformer

outperform tech giant million dollar supercomputer massive dataset train transformer model ai model general key concept mind focus solution like likely sink ship point focus system need like book design explain transformer solution exist market instead book aim explain transformer ecosystem flexible adapt situation face nlp project big tech giant wide range corporate customer use cloud service result add transformer api exist cloud architecture

require effort solution small company individual access powerful transformer model api practically investment development intern implement api day need engineer ph.d.

simple implementation example openai platform saas software service api effective transformer model market openai transformer model effective humanlike present policy require potential user fill request form request accept user access universe natural language processing simplicity openai api take user surprise obtain api key click import openai notebook lin enter nlp task wish prompt receive response completion certain number token length welcome fourth industrial revolution ai 4.0 industry 3.0 developer focus code solution evolve industry 4.0 developer cross disciplinary mindset allennlp offer free use online educational interface transformer allennlp provide library instal notebook example suppose ask im- plement coreference resolution start run example online 4.0 developer learn design way transformer model expect intuitively tell like 3.0

developer explore new approach gpt-2 gpt-3 model chapter 7 rise suprahuman transformers gpt-3 engine transformers coreference resolution task involve find entity word refer sentence show figure 1.5 figure 1.5 run nlp task online word refer website transformer model case bert like model decide link transformer model allennlp provide format output show figure 1.6 figure 1.6 output allennlp transformer model example run https://demo.allennlp.org/coreference-resolution transformer model continuously update obtain different result api satisfy need limit multipurpose api reason- ably good task good specific nlp task translate transformer easy task case 4.0 developer consultant project manager prove api solve specific nlp task require need search solid library choose ready use api drive library book explore library example google advanced ai lab world google trax instal line google colab choose free pay service hand source code tweak model train server google cloud example step ready use api customize transformer model translation task prove educational effective case explore recent evolution google translation implement google trax

chapter 6 machine translation transformer see api openai require limited developer skill library google trax dig bit deeply code approach ai 4.0 api require development editor api effort implement transformer famous online application use transformer algorithm google translate google translate online api let try translate sentence require coreference resolution english french transla- tion google translate figure 1.7 coreference resolution translation google translate google translate appear solve coreference resolution word transformateur french mean electric device word transformer neologism new word french artificial intelligence specialist require language linguistic skill specific project significant development require case project require clarify input request translation example show team linguist acquire linguistic skill work input context addition lot development enhance input interface context hand dirty add script use google translate find transformer model specific translation need bert t5 model explore book choose model easy task increase range

t5 model explore book choose model easy task increase range solution transformer choose transformer model big tech corporation dominate nlp market google facebook microsoft run bil- lion nlp routine day increase ai model unequaled power big giant offer wide range transformer model rank foundation model small company spot vast nlp market enter game hugging face free pay service approach challenge hugging face reach level efficiency acquire billion dollar pour google research lab microsoft funding openai entry point foundation model fully train transformer supercomputer gpt-3 google bert hugging face different approach offer wide range number transformer model task interesting philosophy hugging face offer flexible model addition hugging face offer high level api developer control api explore hugging face chapter book educational tool possible solution specific task openai focus handful potent transformer engine globally perform nlp task human level power openai gpt-3 engine chapter 7 rise suprahuman transformers gpt-3 engines oppose conflicting strategy leave wide range possible implemen- tation

define role industry 4.0 artificial intelligence specialist role industry 4.0 artificial intelligence specialist industry 4.0 connect machine communicate directly machine ai drive iot signal trigger automate decision human inter- vention nlp algorithm send automate report summary email advertisement artificial intelligence specialist adapt new era increasingly automate task include transformer model implementation artificial intelligence specialist new function list transformer nlp task ai specialist appear high level task require little development artificial intelligence specialist ai specialist ai guru provide design idea expla- nation implementation let example api openai api require ai developer web designer

create form linguist subject matter expert sme prepare prompt input text primary role ai specialist require linguistic skill tell gpt-3 engine accomplish task show example involve work context input new task name prompt engineering prompt engineer future ai library google trax library require limited development start ready use model ai specialist master linguistic nlp task work dataset output training fine tuning hugging face functionality require limited development provide api library case hand dirty case training fine tune model find correct hyperparameter require expertise artificial intelligence specialist development level skill project tokenizer dataset match explain chapter 9 match tokenizer dataset case artificial intel- ligence developer work linguist example play crucial role computational linguistics training come handy level recent evolution nlp ai term embed transformer disrupt ai development ecosystem gpt-3 transformer currently embed microsoft azure application github copilot example introduce foundation model section chapter codex example look chapter 16 emergence transform- embed transformer accessible directly provide automatic development support automatic code generation usage embed transformer seamless end user assist text pragmatic definition transformer represent artificial intelligence specialist vary ecosystem transformer explore fascinating new world embed transformer chapter 16 good chapter master previous chapter concept example skillset industry 4.0 ai specialist require flexibility cross disciplinary knowledge flexibility book provide artificial intelligence specialist variety

transformer ecosystem adapt new paradigm market time summarize idea chapter dive fascinating architecture original transformer chapter 2 fourth industrial revolution industry 4.0 force artificial intelligence pro- found evolution industrial revolution digital industry 4.0 build digital revolution connect automated process replace human decision critical area include nlp rnn limitation slow progression automate nlp task require fast mov- e world transformer fill gap corporation need summarization translation wide range nlp tool meet challenge industry 4.0 industry 4.0 i4.0

spur age artificial intelligence industrialization evo- lution concept platform framework language model represent challenge industry 4.0 developer foundation model bridge gap industrial revolution i4.0 provide homogenous model carry wide range task training fine tuning website allennlp example provide educational nlp task installation provide resource implement transformer model customize program openai provide api require code line run powerful gpt-3 engine google trax provide end end library hugging face offer transformer model implementation explore ecosystem book access gpt-3 engine directly create openai account use api directly run example openai user interface industry 4.0 radical deviation ai broad skillset example project manager decide implement transformer ask web designer create interface openai api prompt engineering require project manager ask artificial intelligence specialist download google trax hugging face develop blow project customize transformer model industry 4.0 game changer developer role expand require designing programming addition embedded transformer provide assisted code development usage new skillset challenge open new exciting horizon chapter 2 getting start architecture transformer model start architecture original transformer industrial revolution true false fourth industrial revolution connect true false industry 4.0 developer ai development true false industry 4.0 developer implement transformer scratch true false necessary learn transformer ecosystem hugging face example true false 6 ready use transformer api satisfy need true false company accept transformer ecosystem developer know well true false 8 cloud transformer

mainstream true false 9 transformer project run laptop true false 10 industry 4.0 artificial intelligence specialist flexible true false bommansani et al 2021 opportunities risks foundation models https:// chen et al ,2021 evaluate large language model train code https://arxiv.org/ microsoft ai https://innovation.microsoft.com/en-us/ai-at-scale google ai https://ai.google/ transformer google trax https://github.com/google/trax hugging face https://huggingface.co/

join book discord space join book discord workspace monthly ask session author getting start architecture transformer language essence human communication civilization bear word sequence form language live world digital rep- resentation language daily life rely nlp digitalize language function web search engine email social network post tweet smartphone texting translation web page speech text stream site transcript text speech hotline service everyday function chapter 1 transformer explain limit rnn birth cloud ai trans- former take fair share design development role industry 4.0 devel- oper understand architecture original transformer multiple transformer ecosystem follow december 2017 google brain google research publish seminal vaswani et al atten- tion need paper transformer bear transformer outperform exist state art nlp model transformer train fast previous architecture obtain high evaluation result result transformer key component nlp idea attention head transformer away recurrent neural network feature chapter open hood transformer model describe vaswani et al 2017 examine main component architecture explore fascinating world attention illustrate key component transformer getting start architecture transformer model chapter cover following topic architecture transformer transformer self attention model encoding decoding stack input output embedding let dive directly structure original transformer architecture rise transformer attention december 2017 vaswani et al 2017 publish seminal paper attention need perform work google research google brain refer model describe attention need original transformer model chapter book section look structure transformer model build follow section explore inside component model original transformer model stack 6 layer output layer l input layer l+1 final prediction reach 6 layer encoder stack left 6 layer decoder stack right

appendix terminology transformer models help transition classi- cal usage deep learning word transformer vocabulary appendix summarize change classical ai definition neural network model figure 2.1 architecture transformer left input enter encoder transformer attention sublayer feedforward sublayer right target output decoder trans- attention sublayer feedforward

feedforward sublayer right target output decoder trans- attention sublayer feedforward network sublayer immediately notice rnn lstm cnn recurrence abandon architecture attention replace recurrence function require increase parameter distance be- tween word increase attention mechanism word word operation actually token token operation word level explanation simple attention mechanism find word relate word sequence include word analyze let examine following sequence cat sit mat getting start architecture transformer model attention run dot product word vector determine strong relationship word word include cat cat figure 2.2 attend word attention mechanism provide deep relationship word produce well attention sublayer original transformer model run attention mechanism parallel speed calculation explore architecture following section encoder stack process name multi head attention provide broad depth analysis sequence preclusion recurrence reduce calculation operation implementation parallelization reduce training time attention mechanism learn different perspective input sequence look transformer structure outside let component transformer start encoder encoder stack layer encoder decoder original transformer model stack layer layer encoder stack following structure attention replace recurrence creative aspect transformer critical attention mechanism look inside architecture figure 2.3 layer encoder stack transformer original encoder layer structure remain n=6 layer transformer model layer contain main sublayer multi headed attention mechanism fully con- nected position wise feedforward network notice residual connection surround main sublayer sublayer(x transformer model connection transport unprocessed input x sublayer layer normalization function way certain key information positional encoding lose way normalized

output layer layernormalization x + sublayer(x structure n=6 layer encoder identical content layer strictly identical previous layer example embed sublayer present level stack layer contain embedding layer guarantee encode input stable layer getting start architecture transformer model multi head attention mechanism perform function layer 1 6 how- perform task layer learn previous layer explore different way associate token sequence look association word like look different association letter word solve crossword

association word like look different association letter word solve crossword puzzle designer transformer introduce efficient constraint output sublayer model constant dimension include embedding layer residual connection dimension dmodel set value depend goal original transformer architecture dmodel = 512 dmodel

powerful consequence practically key operation dot product result dimension remain stable reduce number operation calculate reduce machine consumption make easy trace information flow model global view encoder show highly optimize architecture transformer follow section zoom sublayer mechanism begin embedding sublayer input embedding sublayer convert input token vector dimension dmodel = 512 learn embedding original transformer model structure input embedding figure 2.4 input embedding sublayer transformer embedding sublayer work like standard transduction model tokenizer trans- form sentence token tokenizer method bpe word piece sen- tence piece method transformer initially bpe model use method goal similar choice depend strategy choose example tokenizer apply sequence transformer innovative nlp model produce following token type model transform er innovative n l p notice tokenizer normalize string lowercase truncate subpart tokenizer generally provide integer representation embedding process example text = cat sleep couch tired tokenized text= 1996 4937 7771 2006 1996 6411 1012 2009 2001 2205 5458 2000 2131 2039 1012 information tokenize text point tokenize text embed transformer contain learn embedding sublayer embedding method apply tokenize input choose skip gram

architecture word2vec embedding approach google available 2013 illustrate embedding sublayer transformer skip gram focus center word window word predict context word example word(i cen- ter word step window skip gram model

analyze word(i-2 word(i-1 word(i+1 word(i+2 window slide repeat process skip gram model generally contain input layer weight hide layer output contain word embedding tokenize input word suppose need perform embedding follow sentence black cat sit couch brown dog sleep rug focus word black brown word embedding vector word similar getting start architecture transformer model produce vector size dmodel = 512 word obtain size 512 vector embedding word black=[[-0.01206071 0.11632373 0.06206119 0.01403395 0.09541149 0.10695464 0.02560172 0.00185677 -0.04284821 0.06146432 0.09466285 0.04642421 0.08680347 0.05684567 -0.00717266 -0.03163519 0.03292002 -0.11397766 0.01304929 0.01964396 0.01902409 0.02831945 0.05870414 0.03390711 -0.06204525 0.06173197 -0.08613958 -0.04654748 0.02728105 0.04340003 -0.13192849 -0.00945092 -0.00835463 -0.06487109 0.05862355 -0.03407936 -0.00059001 -0.01640179 0.04123065 -0.04756588 0.08812257 0.00200338 -0.0931043 -0.03507337 0.02153351 -0.02621627 -0.02492662 -0.05771535 -0.01164199 -0.03879078 -0.05506947 0.01693138 -0.04124579 -0.03779858 -0.01950983 -0.05398201 0.07582296 0.00038318 -0.04639162 -0.06819214 0.01366171 0.01411388 0.00853774 0.02183574 -0.03016279 -0.03184025 -0.04273562 word black represent 512 dimension embedding method dmodel high number dimension word embedding brown represent 512 dimension brown= 1.35794589e-02 -2.18823571e-02 1.34526128e-02 6.74355254e-02 1.04376070e-01 1.09921647e-02 -5.46298288e-02 -1.18385479e-02 4.41223830e-02 -1.84863899e-02 -6.84073642e-02 3.21860164e-02 4.09143828e-02 -2.74433400e-02 -2.47369967e-02 7.74542615e-02 9.80964210e-03 2.94299088e-02 2.93895267e-02 -3.29437815e-02 7.20389187e-02 1.57317147e-02 -3.10291946e-02 -5.51304631e-02 -7.03861639e-02 7.40829483e-02 1.04319192e-02 -2.01565702e-03 2.43322570e-02 1.92969330e-02 2.57341694e-02 -1.13280728e-01 8.45847875e-02

4.90090018e-03 5.33546880e-02 -2.31553353e-02 3.87288055e-05 3.31782512e-02 -4.00604047e-02 -1.02028981e-01 3.49597558e-02 -1.71501152e-02 3.55573371e-02 -1.77437533e-02 -5.94457164e-02 2.21221056e-02 9.73121971e-02 -4.90022525e-02 verify word embedding produce word use cosine similarity word embedding word black brown similar cosine similarity use euclidean l2 norm create vector unit sphere dot product vector compare cosine point vector theory cosine similarity consult scikit learn documentation source https://scikit-learn.org/stable/modules/metrics.html#cosine-similarity cosine similarity black vector size dmodel = 512 brown vector size dmodel = 512 embedding example cosine_similarity(black brown)= 0.9998901

skip gram produce vector close detect black brown form color subset dictionary word transformer subsequent layer start handed learn word em- bedding provide information word associate big chunk information miss additional vector information indicate word position sequence designer transformer come innovative feature positional encoding let positional encoding work enter positional encoding function transformer idea position word sequence figure 2.5 positional encoding create independent positional vector high cost training speed transformer attention sublayer overly complex work idea add positional encoding value input embedding instead have additional vector describe position token sequence getting start architecture transformer model transformer expect fix size dmodel = 512 constant value model vector output positional encoding function sentence word embedding sublayer black brown semantically similar far apart sentence black cat sit couch brown dog sleep rug word black position 2 pos=2 word brown position 10 pos=10 problem find way add value word embedding word information need add value dmodel = 512 dimension word embedding vector need find way provide information range(0,512 di- mension word embedding vector black brown way achieve positional encoding section focus designer clever way use unit sphere represent positional encoding sine cosine value remain small useful vaswani et al 2017 provide sine cosine function generate different frequen- cie positional encoding pe position

dimension dmodel = 512 word embedding vector (=

(= industry 4.0 pragmatic model agnostic original transformer

model vector contain word embedding position encoding explore disentangled attention separate matrix positional encoding chapter 15 nlp task agnostic transformer models start beginning word embedding vector begin constant 512 i=0 end i=511 mean sine function apply number cosine function odd number implementation differently case domain sine function [0,255 domain cosine function [256,512 produce similar result section use function

way describe vaswani et al 2017 literal translation python pseudo code produce following code

al 2017 literal translation python pseudo code produce following code positional vector pe[0][i position pos range(0 512,2

pe[0][i = math.sin(pos 10000 2 i)/d_model pe[0][i+1 = math.cos(pos 10000 2 i)/d_model go want plot sine function example pos=2 google following plot example enter plot request figure 2.6 plot google google brain trax hugging face provide ready use library word embedding section present positional encoding section need run code share section wish explore code find google colaboratory positional_encoding.ipynb notebook text.txt file chapter github repository getting start architecture transformer model obtain following graph figure 2.7 graph sentence parse section black position pos=2 brown position pos=10 black cat sit couch brown dog sleep rug apply sine cosine function literally pos=2 obtain size=512 positional

9.09297407e-01	-4.16146845e-01	9.58144367e-01	-2.86285430e-01	9.87046242e-01
-1.60435960e-01	9.99164224e-01	-4.08766568e-02	9.97479975e-01	7.09482506e-02
9.84703004e-01	1.74241230e-01	9.63226616e-01	2.68690288e-01	9.35118318e-01
3.54335666e-01	9.02130723e-01	4.31462824e-01	8.65725577e-01	5.00518918e-01
8.27103794e-01	5.62049210e-01	7.87237823e-01	6.16649508e-01	7.46903539e-01
6.64932430e-01	7.06710517e-01	7.07502782e-01	5.47683925e-08	1.00000000e+00
5.09659337e-08	1.00000000e+00	4.74274735e-08	1.00000000e+00	4.41346799e-08

1.00000000e+00 4.10704999e-08 1.00000000e+00 3.82190599e-08 1.00000000e+00 3.55655878e-08 1.00000000e+00 3.30963417e-08 1.00000000e+00 3.07985317e-08 1.00000000e+00 2.86602511e-08 1.00000000e+00 2.66704294e-08 1.00000000e+00 2.48187551e-08 1.00000000e+00 2.30956392e-08 1.00000000e+00 2.14921574e-08 1.00000000e+00 obtain size=512 positional encoding vector position 10 pos=10 -5.44021130e-01 -8.39071512e-01 1.18776485e-01 -9.92920995e-01 6.92634165e-01 -7.21289039e-01 9.79174793e-01 -2.03019097e-01 9.37632740e-01 3.47627431e-01 6.40478015e-01 7.67976522e-01 2.09077001e-01 9.77899194e-01 -2.37917677e-01 9.71285343e-01 -6.12936735e-01 7.90131986e-01 -8.67519796e-01 4.97402608e-01 -9.87655997e-01 1.56638563e-01 -9.83699203e-01 -1.79821849e-01 2.73841977e-07 1.00000000e+00 2.54829672e-07 1.00000000e+00 2.37137371e-07 1.00000000e+00 2.20673414e-07 1.00000000e+00 2.05352507e-07 1.00000000e+00 1.91095296e-07 1.00000000e+00 1.77827943e-07 1.00000000e+00 1.65481708e-07 1.00000000e+00 1.53992659e-07 1.00000000e+00 1.43301250e-07 1.00000000e+00 1.33352145e-07 1.00000000e+00 1.24093773e-07 1.00000000e+00 1.15478201e-07 1.00000000e+00 1.07460785e-07 1.00000000e+00 look result

obtain intuitive literal translation vaswani et al 2017 function python like check result meaningful cosine similarity function word embedding come handy have well vi- sualization proximity position cosine_similarity(pos(2 pos(10))= 0.8600013 similarity position word black brown lexical field group word similarity different cosine_similarity(black brown)= 0.9998901 encoding position show low similarity value word embedding similarity positional encoding take word apart bear mind word embedding vary corpus train problem add positional encoding word embedding vector getting start architecture transformer model add positional encoding embedding vector author transformer find simple way merely add positional encoding vector word embedding vector figure 2.8 positional encoding word embedding black example y1 = black ready add positional vector pe(2 obtain positional encoding function

obtain positional encoding pc(black input word black pc(black = y1 + pe(2 solution straightforward apply show lose information word embedding minimize positional encoding vector possibility increase value y1 sure information word embedding layer efficiently subsequent layer possibility add arbitrary value y1 word embedding black y1 math.sqrt(d_model add positional vector embedding vector word black size 512 range(0 512,2 pe[0][i = math.sin(pos 10000 2 i)/d_model pc[0][i = y[0][i]*math.sqrt(d_model))+ pe[0][i pe[0][i+1 = math.cos(pos 10000 2 i)/d_model pc[0][i+1 = y[0][i+1]*math.sqrt(d_model))+ pe[0][i+1 result obtain final positional encoding vector dimension dmodel = 512 9.09297407e-01 -4.16146845e-01 9.58144367e-01 -2.86285430e-01 9.87046242e-01 -1.60435960e-01 9.99164224e-01 -4.08766568e-02 4.74274735e-08 1.00000000e+00 4.41346799e-08 1.00000000e+00 4.10704999e-08 1.00000000e+00 3.82190599e-08 1.00000000e+00 2.66704294e-08 1.00000000e+00 2.48187551e-08 1.00000000e+00 2.30956392e-08 1.00000000e+00 2.14921574e-08 1.00000000e+00 operation apply word brown word sequence apply cosine similarity function positional encoding vector black brown cosine_similarity(pc(black pc(brown))= 0.9627094 clear view positional encoding process cosine similarity function apply state represent word black brown 0.99987495 word similarity 0.8600013 positional encoding vector similarity 0.9627094

final positional encoding similarity see initial word similarity embedding high value 0.99 see positional encoding vector position 2 10 draw word apart low similarity value 0.86 finally add word embedding vector word respective positional encoding vector see bring cosine similarity word 0.96 getting start architecture transformer model positional encoding word contain initial word embedding information positional encoding value output positional encoding lead multi head attention sublayer sublayer 1 multi head attention multi head attention sublayer contain head follow post layer normaliza- tion add residual connection output sublayer normalize figure 2.9 multi head attention sublayer section begin architecture attention layer example multi atten- tion implement small module python finally post layer normalization

describe let start architecture multi head attention architecture multi head attention input multi attention sublayer layer encoder stack vector contain embedding positional encoding word layer stack start operation dimension vector word xn input sequence dmodel = 512 -01 d512=1.00000000e+00 representation word xn vector dmodel = 512 dimension word map word determine fit sequence follow sentence relate cat rug sequence sequence = cat sit rug dry clean model train find relate cat rug run huge calculation train model dmodel = 512 dimension point view time analyze sequence dmodel block furthermore calculation time find perspective well way divide dmodel = 512 dimension word xn x word sequence 8 dk=64 dimension run 8 head parallel speed training obtain 8 different repre- sentation subspace word relate figure 2.10 multi head representation 8 head run parallel head decide fit cat fit rug rug fit dry clean output head matrix zi shape x dk output multi attention head z define z = z0 z1 z2 z3 z4 z5 z6 z7 z concatenate output multi head sublayer sequence dimension line xm dmodel matrix exit multi head attention sublayer element z concatenate multihead(output = concat(z0 z1 z2

attention sublayer element z concatenate multihead(output = concat(z0 z1 z2 z3 z4 z5 z6 z7 = x dmodel notice head concatenate z dimension dmodel = 512 output multi headed layer respect constraint original transformer model inside head hn attention mechanism word matrix representation query matrix q dimension dq = 64 seek key value pair word matrix key matrix k dimension dk = 64 train provide at- get start architecture transformer model value matrix v dimension dv = 64 train provide attention define scale dot product attention represent follow equation plug q k v (= matrix
dimension make relatively simple use scale dot product obtain attention value head concatenate output z 8 head obtain q k v train model weight matrix qw kw vw dk = 64 column dmodel = 512 row example q obtain dot product x qw q dimension dk = 64

google brain trax openai hugging face provide ready use library book let open hood

transformer model hand dirty python illustrate architecture explore visualize model code in- use basic python code numpy softmax function 10 step run key aspect attention mechanism let start build step 1 model represent input modify parameter number layer head dmodel dk variable transformer fit model chapter describe orig- inal transformer parameter vaswani et al 2017 essential understand original architecture modify explore variant original model design bear mind industry 4.0 developer face challenge multiple ar- chitecture algorithm step 1 represent input save multi_head_attention_sub_layer.ipynb google drive sure gmail account open google colaboratory notebook github repository start minimal python function understand transformer low level inner working attention head explore inner working multi head attention sublayer basic code

import numpy np scipy.special import softmax input attention mechanism build scale dmodel = = 4 instead dmodel = 512 bring dimension vector input x dmodel = 4 easy x contain 3 input 4 dimension instead 512 print("step 1 input 3 input d_model=4 x = np.array([[1.0 0.0 1.0 0.0 input 1 0.0 2.0 0.0 2.0 input 2 1.0 1.0 1.0 1.0 input 3 output show 3 vector dmodel = 4 step 1 input 3 input d_model=4 1 0 1 0 0 2 0 2 1 1 1 1 step model ready figure 2.11 input multi head attention sublayer add weight matrix model getting start architecture transformer model step 2 initialize weight matrix input 3 weight matrix qw train query kw train key vw train value 3 weight matrix apply input model weight matrix describe vaswani et al 2017 dk = = 64 dimension let scale matrix dk = =3 dimension scale 3 4 weight matrix able visualize intermediate result easily perform dot product input x. weight matrix initialize start query weight matrix print("step 2 weight 3 dimension x d_model=4 w_query = np.array([[1 0 1 1 0 0 0 0 1 0 1 1 output w_query weight matrix 1 0 1 1 0 0 0 0 1 0 1 1 initialize key weight matrix w_key = np.array([[0 0 1 1 1 0 0 1 0 size shape matrix educational notebook arbitrary goal overall process attention mechanism 1 1 0

output key weight matrix 0 0 1 1 1 0 0 1 0 1 1 0 finally initialize value weight matrix w_value =

np.array([[0 2 0 0 3 0 1 0 3 1 1 0 output value weight matrix 0 2 0 0 3 0 1 0 3 1 1 0 second step model ready figure 2.12 weight matrix add model multiply weight input vector obtain q k v. step 3 matrix multiplication obtain q k v multiply input vector weight matrix obtain query key value vector input getting start architecture transformer model model assume w_query w_key w_value weight matrix input approach possible let multiply input vector w_query weight matrix print("step 3 matrix multiplication obtain q k v print("query x w_query output vector q1 = = 64= 1 0 2 q2= 2,2 2 q3= 2,1 3 step 3 matrix multiplication obtain q k v query x w_query 1 0 2 2 2 2 2 1 3 multiply input vector w_key weight matrix print("key x w_key obtain vector k1= 0 1 1 k2= 4 4 0 k3= 2 3 1 key x w_key 0 1 1 4 4 0 2 3 1 finally multiply input vector w_value weight matrix print("value x w_value obtain vector v1= 1 2 3 v2= 2 8 0 v3= 2 6 3

value x w_value 1 2 3 2 8 0 2 6 3 step model ready figure 2.13 q k v generate q k v value need calculate attention score step 4 scale attention score attention head implement original transformer equation (= step 4

focus q k model round = 3 = 1.75 1 plug value q k print("step 4 scale attention score k_d=1 square root k_d=3 round 1 example attention_score = q k.transpose())/k_d intermediate result display step 4 scale attention scores 2 4 4 4 16 12 4 12 10

getting start architecture transformer model step 4 complete example score x1 2,4,4 k vector head figure 2.14 scale attention score input 1 attention equation apply softmax intermediate score vector step 5 scale softmax attention score vector multiplication let zoom individual vector print("step 5 scale softmax attention_score vector obtain scale softmax attention score vector step 5 scale softmax attention_score vector 0.06337894 0.46831053 0.46831053 6.03366485e-06 9.82007865e-01 1.79861014e-02 2.95387223e-04 8.80536902e-01 1.19167711e-01 step 5 complete example softmax score x1 key figure 2.15 softmax score input 1 key calculate final attention value complete equation step 6 final attention representation finalize attention equation plug v (=

calculate attention score input x1 step 6 7 calculate attention value

word vector reach step 8 generalize attention calculation input vector obtain attention(q k v x1 multiply intermediate attention score 3 value vector zoom inner working equation print("step 6 attention value obtain score1 k_d v get start architecture transformer model step 6 attention value obtain score1 k_d v 1 2 3 2 8 0 2 6 3 0.06337894 0.12675788 0.19013681 0.93662106 3.74648425 0 0.93662106 2.80986319 1.40493159 step 6 complete example 3 attention value x1 input calculate figure 2.16 attention representation attention value need sum getting start architecture transformer model step 7 sum result 3 attention value input 1 obtain sum obtain line print("step 7 sum result create line output output line output matrix input 1 step 7 sum result create line output matrix 1.93662106 6.68310531 1.59506841 second line output input input 2 example sum attention value x1 figure 2.17 figure 2.17

sum result input complete step input 1 need add result input step 8 step 1 7 input transformer produce attention value input 2 input 3 method describe step 1 step 7 attention head step onwards assume 3 attention value learn weight dmodel = 64 want original dimension look like reach sub- see attention representation process detail small model let directly result assume generate 3 attention representation dimension dmodel = 64 print("step 8 step 1 7 input 1 3 assume 3 result learn weight train example assume implement original transformer paper 3 result 64 dimension follow output display simulation z0 represent 3 output vector dmodel = 64 dimension head 1 step 8 step 1 7 input 1 3 0.31982626 0.99175996 (61 squeeze values) 0.16233212 0.99584327 0.55528662 (61 squeeze values) 0.70160307 0.14811583 0.50875291 (61 squeeze values) 0.83141355 result vary run notebook stochastic nature gener- ation vector transformer output vector input head step gen- erate output 8 head create final output attention sublayer getting

start architecture transformer model step 9 output head attention sublayer assume train 8 head attention sublayer transformer 3 output vector 3 input vector word word piece dmodel =

64 dimension print("step 9 assume train 8 head attention print("shape head",z0h1.shape,"dimension 8 heads",64 8) output show shape head step 9 assume train 8 head attention sublayer shape head 3 64 dimension 8 head 512 8 head produce z z = z0 z1 z2 z3 z4 z5 z6 z7 transformer concatenate 8 element z final output multi head step 10 concatenation output head transformer concatenate 8 element z multihead(output = concat z0 z1 z2 z3 z4 z5 z6 z7 w 0 = x dmodel note z multiply w 0 weight matrix train model assume w 0 train integrate concatenation function z0 z7 concantenate print("step 10 concantenation head 1 8 obtain original 8x64=512 ouput dimension model output concatenation z step 10 concatenation head 1 8 obtain original 8x64=512 output dimension model 0.65218495 0.11961095 0.9555153 0.48399266 0.80186221 0.16486792 0.95510952 0.29918492 0.7010377 0.20682832 0.4123836 0.90879359 0.20211378 0.86541746 0.01557758 0.69449636 0.02458972 0.889699 concatenation visualize stack element z figure 2.18 attention sublayer output concatenation produce standard dmodel = 512

dimensional output figure 2.19 concatenation output 8 head layer normalization process attention sublayer attention sublayer feedforward sublayer transformer follow post lay- er normalization post ln figure 2.20 post layer normalization getting start architecture transformer model post ln contain add function layer normalization process add function pro- cesse residual connection come input sublayer goal residual connection sure critical information lose post ln layer normalization describe follow layernormalization x + sublayer(x sublayer(x sublayer x information available input step sublayer(x input layernormalization vector v result x + sublayer(x dmodel = 512 input output transformer standardize process layer normalization method exist variation exist model basic concept v = x + sublayer(x define layernormalization v (= variable

mean v dimension d. standard deviation v dimension d. scale parameter bias vector

version layernormalization v show general idea possible post ln meth- od sublayer process output post ln layernormalization v case sublayer feedforward network sublayer 2

feedforward network input feedforward network ffn dmodel = 512 output post ln pre- figure 2.21 feedforward sublayer ffn sublayer describe follow ffn encoder decoder fully connect ffn position wise network position process separately iden- ffn contain layer apply relu activation function input output ffn layer dmodel = 512 inner layer large dff ffn view perform convolution size 1 kernel take description account describe optimize standardized ffn ffn(x = max 0 xw1 + b1 w2+b2 output ffn go post ln describe previous section output send layer encoder stack multi head attention layer decoder stack let explore decoder stack get start architecture transformer model decoder stack layer decoder transformer model stack layer like encoder layer layer decoder stack following structure figure

2.22 layer decoder stack transformer structure decoder layer remain encoder n=6 layer transformer model layer contain sublayer multi headed mask attention mech- anism multi headed attention mechanism fully connect position wise feedforward decoder main sublayer mask multi head attention mechanism sublayer output give position following word mask transformer base assumption inference see rest sequence way model future part sequence residual connection sublayer(x surround main sublayer transformer model like encoder stack layernormalization x + sublayer(x embed layer sublayer present level stack like encoder stack output sublayer decoder stack constant dimension dmodel like encoder stack include embedding layer output residual connection designer work hard create symmetrical encoder decoder stack structure sublayer function decoder similar encoder section refer encoder functionality need focus difference decoder encoder output embedding position encoding structure sublayer decoder sublayer encoder output embedding layer position encoding function encoder stack transformer usage explore model present vaswani et al 2017 output translation need learn choose use french translation output = le chat noir était assis sur le canapé et le chien marron dormait sur le tapis output french translation

english input sentence input = black cat sit couch brown dog sleep rug output word word

embedding layer positional encoding func- tion like layer encoder stack let specific property multi head attention layer decoder stack attention layer transformer auto regressive model use previous output sequence addi- tional input multi head attention layer decoder use process encoder getting start architecture transformer model masked multi head attention sublayer 1 let attention apply position include current position future word hide transformer force learn predict post layer normalization process follow masked multi head attention sublayer 1 multi head attention sublayer 2 attend position current position transformer predict avoid see sequence predict multi head attention sublayer 2 draw information encoder take encoder k v account dot product attention operation sublayer draw in- formation masked multi head attention sublayer 1 masked attention take sublayer 1(q account dot product attention operation decoder use train information encoder define input self attention multi head sublayer decoder input_attention = output_decoder_sub_layer 1(q output_encoder_layer(k v post layer normalization process follow mask multi head attention sublayer 1 transformer go ffn sublayer follow post ln linear layer ffn sublayer post ln linear layer ffn sublayer structure ffn encoder stack post layer normal- ization ffn work layer normalization encoder stack transformer produce output sequence element time output sequence = y1 y2 yn linear layer produce output sequence linear function vary model rely standard method y = w x + b w b learn parameter linear layer produce probable element sequence softmax func- tion convert probable element decoder layer like encoder layer layer l layer l+1 layer n=6 layer transformer stack let transformer train performance obtain training performance original transformer train 4.5 million sentence pair english german dataset 36 million sentence pair english french dataset dataset come workshops machine translation wmt find following link wish explore

come workshops machine translation wmt find following link wish explore wmt dataset http://www.statmt.org/wmt14/ training original transformer base model take 12 hour train 100,000 step machine 8 nvidia p100 gpu big model take 3.5 day 300,000 step original

transformer outperform previous machine translation model bleu score 41.8 result obtain wmt english french dataset bleu stand bilingual evaluation understudy algorithm evaluate quality result machine translation google research google brain team apply optimization strategy improve per- formance transformer example adam optimizer learning rate vary go warmup state linear rate decrease rate afterward different type regularization technique residual dropout dropout apply sum embedding transformer apply label smoothing avoid overfitting overconfident hot output introduce accurate evaluation force model train well transformer model variation lead model usage explore subsequent chapter end chapter let feel simplicity ready use transformer model hugging face example tranformer model hugging face see chapter condense ready use hugging face trans- getting start architecture transformer model hugging face implement machine translation line code open multi_head_attention_sub_layer.ipynb google colaboratory save notebook google drive sure gmail account cell ensure hugging face transformer instal pip -q install transformer cell import hugging face pipeline contain transformer usage @title retrieve pipeline module choose english french transformer import pipeline implement hugging face pipeline contain ready use function case illustrate transformer model chapter activate translator model enter sentence translate english french translator = pipeline("translation_en_to_fr line code print(translator("it easy translate language transformer voilà translation display translation_text il est facile de traduire des langues à l'aide de hugging face show transformer architecture ready use model chapter get start examine mind blow long distance dependency transformer architecture uncover transformer perform transduction write oral sequence meaningful representation history natural language dimension expansion transduction simplification implementation take artificial intelligence level see explore bold approach remove rnns lstms cnn transduction prob- lem sequence modeling build transformer architecture

lstms cnn transduction prob- lem sequence modeling build transformer architecture

symmetrical design standardized dimension encoder decoder make flow sublayer nearly seamless see remove recurrent network model transformer introduce parallelized layer reduce training time discover innovation positional encoding mask multi headed attention flexible original transformer architecture provide basis innovative vari- ation open way powerful transduction problem language modeling zoom depth aspect transformer architecture follow chapter describe variant original model arrival transformer mark beginning new generation ready use artificial intelligence model example hugging face google brain artificial intelligence easy implement line code chapter fine tune bert models explore powerful evolution original transformer model nlp transduction encode decode text representation true false natural language understanding nlu subset natural language processing language modeling algorithm generate probable sequence word base input transformer customize lstm cnn layer true false 5 transformer contain lstm cnn layer true false 6 attention examine token sequence true false transformer use positional vector positional encoding true false 8 transformer contain feedforward network true false mask multi headed attention component decoder transformer prevent algorithm parse give position see rest sequence 10 transformer analyze long distance dependency well lstm true false getting start architecture transformer model ashish vaswani noam shazeer niki parmar jakob uszkoreit llion jones aidan n. go- mez lukasz kaiser illia polosukhin 2017 attention need https://arxiv.org/

hug face transformer usage https://huggingface.co/transformers/usage.html tensor2tensor t2 t introduction https://colab.research.google.com/github/ manuel romero notebook link explanation raimi karim https://colab google language research https://research.google/teams/language/ hugging face research https://huggingface.co/transformers/index.html annotated transformer http://nlp.seas.harvard.edu/2018/04/03/attention.html

jay alammar illustrated transformer http://jalammar.github.io/illustrated- join book discord

space join book discord workspace monthly ask session author fine tune bert model chapter 2 getting start architecture transformer model define building block architecture original transformer think original transformer model build lego ® brick construction set contain brick encoder decoder embedding layer positional encoding method multi head attention layer mask multi head attention layer post layer normalization feed forward sub layer linear output layer brick come size form spend hour build sort model building kit construction require brick con- struction add new piece like obtain additional brick model build bert add new piece transformer building kit bidirectional multi head attention sub layer human problem understand sentence look past word bert like look word sentence time chapter explore architecture bidirectional encoder representations transformers bert bert use block encoder transformer novel way use decoder stack fine tune pretrained bert model bert model fine tune train party upload hugging face transformer pretraine pretrained bert example fine tune nlp task fascinating experience downstream transformer usage hugging face module fine tune bert model chapter cover following topic bidirectional encoder representation transformers bert architecture bert step bert framework prepare pretraine environment define pretraine encoder layer build fine tune bert model load acceptability judgment dataset create attention mask bert model configuration measure performance fine tune model step explore background bert model architecture bert bert introduce bidirectional attention transformer model bidirectional attention require change original transformer model building block transformer describe chapter 2 getting start architecture transformer model consult chapter 2 time review aspect building block transformer section focus specific aspect bert model focus evolution design devlin et al 2018 describe encoder stack encoder stack preparation pretraine input envi- ronment describe step framework bert pretraining fine tuning let explore encoder stack encoder stack building block original transformer model encoder layer

stack encoder stack building block original transformer model encoder layer encoder layer

describe chapter 2 getting start architecture transformer model show figure 3.1 figure 3.1 encoder layer bert model use decoder layer bert model encoder stack decoder stack mask token hide token predict attention layer encoder zoom bert encoder layer follow section original transformer contain stack n=6 layer number dimension original transformer dmodel = 512 number attention head original transformer a=8 dimension head original transformer bert encoder layer large original transformer model bert model build encoder layer bertbase contain stack n=12 encoder layer dmodel = 768 ex- press h=768 bert paper multi head attention sub layer contain a=12 head dimension head za remain 64 original transformer model fine tune bert models output multi head attention sub layer concatenation output 12 head output_multi head_attention={z0 z1 z2 z11

bertlarge contain stack n=24 encoder layer dmodel = 1024 multi head attention sub layer contain a=16 head dimension head za remain 64 original transformer model output multi head attention sub layer concatenation out- 16 head output_multi head_attention={z0 z1 z2 z15 size model sum follow figure 3.2 transformer model bert model limit configuration configura- tion illustrate main aspect bert model numerous variation possible size dimension play essential role bert style pretraining bert model like hu- mans bert model produce well result work memory dimension knowledge datum large transformer model learn large amount datum pretrain well downstream nlp task let sub layer fundamental aspect input embedding positional encoding bert model prepare pretraine input environment bert model decoder stack layer mask multi head attention sub layer bert designer state mask multi head attention layer mask rest sequence impede attention process mask multi head attention layer mask token present position example following sentence cat sit nice rug reach word input encoder cat sit < mask sequence > motivation approach prevent model see output suppose predict left right approach produce relatively good result model learn way know refer need sentence reach word rug figure rug author bert come idea pretrain model prediction different approach author bert come

bidirectional attention let attention head attend word left right right left word self attention mask encoder job hinder mask multi head attention sub layer decoder model train task method masked language modeling mlm second method sentence prediction nsp let start mask language modeling fine tune bert models masked language modeling masked language modeling require train model sequence visible word follow mask sequence predict bert introduce bidirectional analysis sentence random mask word important note bert apply wordpiece subword segmentation tokenization method input use learn positional encoding sine cosine approach potential input sequence cat sit nice rug decoder mask attention sequence model reach word cat sit < mask sequence > bert encoder mask random token prediction cat sit mask nice rug multi attention sub layer sequence run self attention process predict mask token input token mask tricky way force model train long produce well result method surprise model mask single token 10 dataset example cat sit nice rug surprise model replace token random token 10

dataset cat sit nice rug replace token mask token 80 dataset example cat sit mask nice rug author bold approach avoid overfitting force model train efficiently bert train perform sentence prediction sentence prediction second method find train bert sentence prediction nsp input contain sentence 50 case second sentence actual second sentence doc- ument 50 case second sentence select randomly relation new token add cls binary classification token add beginning sequence predict second sequence follow sequence positive sample usually pair consecutive sentence take dataset negative sample create sequence different document sep separation token signal end sequence example input sentence take book cat sleep rug like sleep day sentence complete input sequence cls cat sleep rug sep like sleep e day[sep approach require additional encoding information distinguish sequence sequence embedding process obtain figure 3.3 input embedding input embedding obtain sum token embedding segment sentence phrase word embedding positional encoding embedding fine tune bert

model input embedding positional encoding sub layer bert model sum sequence word break wordpiece token mask token randomly replace initial word token mask language mod- cls classification token insert beginning sequence classification sep token separate sentence segment phrase nsp training sentence embedding add token embedding sentence different sentence embedding value sentence b. positional encoding learn sine cosine positional encoding method original transformer apply additional key feature bert use bidirectional attention multi head attention sub layer open vast horizon learn understand relationship token bert introduce scenario unsupervised embedding pretraine model unla- beled text unsupervised scenario force model think hard multi head attention learning process make bert learn language build apply knowledge downstream task have pretrain time bert use supervise learning cover basis pretraining process bert improve training environment transformer let motivation pretraine help fine tuning process pretraine fine tune bert model bert step framework step pretraine second fine tuning show

model bert step framework step pretraine second fine tuning show figure 3.4 figure 3.4 bert framework train transformer model hour day take time engineer architecture parameter select proper dataset train transformer model fine tune bert model pretraining step bert framework break sub step define model architecture number layer number head dimension building block model train model mlm nsp task second step bert framework fine tuning break initialize downstream model choose train parameter pretrained fine tune parameter specific downstream task recognize textual entailment rte question answering squad v1.1 squad v2.0 situation adversarial generations swag section cover information need fine tune bert model follow chapter explore topic bring section depth

chapter 4 pretraine roberta model scratch pretrain bert like model scratch 15 step compile datum train tokenizer train model chapter aim specific building block bert fine tune exist model chapter 5 downstream nlp tasks transformers down- stream task explore glue squad

v1.1 squad swag nlp evaluation dataset run downstream transformer model illustrate key task goal chapter fine tune downstream model chapter 7 rise suprahuman transformers gpt-3 engines explore architecture unleash usage openai gpt-2 gpt-3 transformer bertbase configure close openai gpt produce well performance openai transformer evolve reach suprahuman nlp level chapter bert model fine tune train corpus linguistic acceptability cola downstream task base neural network acceptability judgments alex warstadt amanpreet singh samuel r. bowman fine tune bert model determine grammatical acceptability sentence fine tune model acquire certain level linguistic competence go bert architecture pretraine fine tune framework let fine tune bert model section fine tune bert model predict downstream task acceptability judgments measure prediction matthews correlation coefficient mcc explain evaluate matthews correlation coefficient section chapter open bert_fine_tuning_sentence_classification_gpu.ipynb google colab sure email account notebook chapter03 github repository book title cell notebook close title subsection chapter examine transformer model hardware constraint account transformer model require multiprocesse hardware runtime menu google co- lab select change runtime type select gpu hardware accelerator drop list transformer model hardware drive recommend read appendix ii hardware constraints transformer models continue chapter program hugging face module install instal hugging face pytorch interface bert hugging face provide pretraine bert model hugging face develop base class name pretrainedmodel instal class load model pretrained model config- hugging face provide module tensorflow pytorch recommend developer comfortable environment excellent ai research team use environment chapter install module require follow @title instal hugging face pytorch interface bert pip install -q transformer installation run requirement

satisfy message display import module need program fine tune bert model import module import pretrained module require pretrained bert tokenizer configuration bert model bertadam optimizer import sequence @title import module import torch.nn nn torch.utils.data

import tensordataset dataloader randomsampler keras.preprocessing.sequence import pad_sequences sklearn.model_selection import train_test_split transformer import berttokenizer bertconfig transformer import adamw bertforsequenceclassification get_linear nice progress bar module import tqdm tqdm import tqdm trange import widely standard python module import pandas pd import numpy np import matplotlib.pyplot plt matplotlib inline go message display bear mind google colab pre instal module vm specify cuda device torch specify torch use compute unified device architecture cuda parallel computing power nvidia card work multi head attention model @title harware verification device attribution device = torch.device("cuda torch.cuda.is_available cpu output vary google colab configuration appendix ii hardware constraint transformer models explanation screenshot load dataset load dataset load cola base warstadt et al 2018 paper general language understanding evaluation glue consider linguistic acceptability priority nlp task chapter 5 downstream nlp tasks transformers explore key task transformer perform prove efficiency following cell notebook automatically download necessary file curl -l https://raw.githubusercontent.com/denis2054/transformers-for- nlp-2nd edition master chapter03 in_domain_train.tsv --output in_domain curl -l https://raw.githubusercontent.com/denis2054/transformers-for- nlp-2nd edition master chapter03 out_of_domain_dev.tsv --output out_of appear file manager figure 3.5 upload dataset program load dataset @title load dataset source dataset https://nyu-mll.github.io/cola/

df = pd.read_csv("in_domain_train.tsv delimiter='\t header = names=['sentence_source label label_notes sentence fine tune bert model output display shape dataset import 10 line sample display visualize acceptability judgment task sequence make sense output show 10 line label dataset change run say tom pay pay like cabbage fred like egg wendy mother country iceland john want win find bug bed girl meet departmental party peter old pig frank promise man leave ve see coward frank drive way buenos aires sample .tsv file contain tab separate column

column 1 source sentence code column 2 label 0 = unacceptable 1 = acceptable column 3 label annotate author column 4 sentence classify open .tsv file locally read sample dataset program process datum bert model create sentence label list add bert token program create sentence describe prepare pretraine input environment section chapter create sentence label list add bert token sentence = df.sentence.values add cls sep token beginning end sentence sentence = cls + sentence + sep sentence sentence label = df.label.values cls sep add program activate tokenizer activate bert tokenizer section initialize pretrained bert tokenizer save time train scratch program select uncased tokenizer activate display tokenize sentence @title activate bert tokenizer tokenizer = berttokenizer.from_pretrained('bert base uncased do_lower tokenized_texts = tokenizer.tokenize(sent send sentence print tokenize sentence output contain classification token sequence segmentation token tokenize sentence cls friend wo n t buy analysis let propose program process datum fine tune bert model process datum need determine fix maximum length process datum model sentence dataset short sure program set maximum length sequence 128 sequence padded @title process datum set maximum sequence length long sequence training set 47 leave room end original paper author length

512 max_len = 128 use bert tokenizer convert token index number bert vocabulary input_id = tokenizer.convert_tokens_to_ids(x x tokenized_texts pad input token input_id = pad_sequences(input_id maxlen = max_len dtype="long sequence process program create attention mask create attention mask come tricky process pad sequence previous cell want prevent model perform attention padded token idea apply mask value 1 token 0 follow padding @title create attention mask attention_mask = create mask 1 token follow 0 padding seq input_id seq_mask = float(i>0 seq program split datum split datum training validation set program perform standard process split datum training validation @title split datum train validation set use train_test_split split datum train validation set train_input validation_input train_label validation_label = train test_split(input_id label random_state=2018 test_size=0.1 train_mask validation_mask = train_test_split(attention_masks data ready train need adapt

torch convert datum torch tensor fine tune model use torch tensor program convert datum torch tensor @title convert datum torch tensor torch tensor require datatype model train_input = torch.tensor(train_inputs validation_input = torch.tensor(validation_inputs train_label = torch.tensor(train_labels validation_label = torch.tensor(validation_labels train_mask = torch.tensor(train_masks validation_mask = torch.tensor(validation_masks conversion need create iterator select batch size create iterator cell program select batch size create iterator iterator clever way avoid loop load datum memory iterator couple torch dataloader batch train massive dataset crash machine memory model batch size 32 @title select batch size create iterator select batch size training fine tune bert specific task author recommend batch size 16 32 batch_size = 32

create iterator datum torch dataloader help save memory training unlike loop iterator entire dataset need load train_data = tensordataset(train_inputs train_mask train_label train_sampler = randomsampler(train_data train_dataloader = dataloader(train_data sampler = train_sampler batch fine tuning bert models validation_data = tensordataset(validation_inputs validation_mask validation_sampler = sequentialsampler(validation_data validation_dataloader = dataloader(validation_data sampler = validation data process set program load configure bert bert model configuration program initialize bert uncased configuration @title bert model configuration initialize bert bert base uncase style configuration @title transformer installation pip -qq install transformer transformer import bertmodel bertconfig configuration = bertconfig initialize model bert base uncase style configuration model = bertmodel(configuration access model configuration configuration = model.config output display main hugging face parameter similar follow library let main parameter attention_probs_dropout_prob 0.1 apply 0.1 dropout ratio attention prob- hidden_act gelu non linear activation function encoder gaussian error linear units activation function input weight magnitude make non linear hidden_dropout_prob 0.1 dropout probability apply fully connect lay- er connection find embedding encoder pooler layer output good reflection content

sequence pool sequence hide state improve output sequence hidden_size 768 dimension encode layer pooler layer initializer_range 0.02 standard deviation value initialize weight intermediate_size 3072 dimension feed forward layer encoder layer_norm_eps 1e-12 epsilon value layer normalization layer max_position_embeddings 512 maximum length model use model_type bert model num_attention_heads 12 number head num_hidden_layers 12 number layer pad_token_id 0 i d padding token avoid training padding token type_vocab_size 2 size token_type_id identify sequence example dog[sep cat.[sep represent token id 0,0,0 fine tuning bert models vocab_size 30522 number different token model represent parameter mind load pretrained model load hugging face bert uncased base model program load pretrained bert model @title load hugging face bert uncased base model model = bertforsequenceclassification.from_pretrained("bert base uncased model = nn

dataparallel(model define model define parallel processing send model device explanation appendix ii hardware constraints transformer models pretraine model train necessary interesting explore archi- tecture detail visualize parameter sublayer show follow excerpt word_embedding embedding(30522 768 padding_idx=0 position_embedding embedding(512 768 token_type_embeddings embedding(2 768 dropout dropout(p=0.1 inplace = false query linear(in_features=768 out_features=768 key linear(in_features=768 out_features=768 bias = true value linear(in_features=768 out_features=768 dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=768 dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=3072 bias = true dense linear(in_features=3072 out_features=768 bias = true dropout dropout(p=0.1 inplace = false query linear(in_features=768 out_features=768 key linear(in_features=768 out_features=768 bias = true value linear(in_features=768 out_features=768 dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=768 dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=3072 bias = true fine tune bert model dense linear(in_features=3072 out_features=768 bias = true dropout dropout(p=0.1 inplace = false let

main parameter optimizer optimizer group parameter program initialize optimizer model parameter fine tune model begin initialize pretraine model parameter value name parameter optimizer include weight decay rate avoid overfitting pa- rameter filter goal prepare model parameter training loop @title optimizer grouped parameter code take apply weight decay parameter name include bert gamma beta parameter

bias param_optimizer = list(model.named_parameters no_decay = bias layernorm.weight separate weight parameter bias parameter weight parameter specify weight_decay_rate bias parameter weight_decay_rate 0.0 optimizer_grouped_parameter = filter parameter include bias gamma param p n p param_optimizer any(nd n nd filter parameter include param p n p param_optimizer any(nd n nd note optimizer_grouped_parameters include parameter value name parameter prepare clean ready training loop hyperparameter training loop hyperparameter training loop critical innocuous adam activate weight decay warm phase example learning rate lr warm rate warmup set small value early optimization phase gradually increase certain number iteration avoid large gradient overshoot optimization goal researcher argue gradient output level sub layer layer nor- malization require warm rate solve problem require experimental run optimizer bert version adam call bertadam @title hyperparameter training loop optimizer = bertadam(optimizer_grouped_parameters program add accuracy measurement function compare prediction label create accuracy measurement function function calculate accuracy prediction vs label def flat_accuracy(preds label pred_flat = np.argmax(preds axis=1).flatten labels_flat = labels.flatten return np.sum(pred_flat = = labels_flat len(labels_flat data ready parameter ready time activate training loop training loop training loop follow standard learning process number epoch set 4 measurement loss accuracy plot training loop use dataloader load train batch training process measure evaluate fine tune bert model code start initialize train_loss_set store loss accuracy plot start train epoch run standard training loop show @title training loop t = store loss accuracy plot train_loss_set = number training epoch author recommend 2 4 epoch = 4 trange tqdm

wrapper normal python range trange(epoch desc="epoch tmp_eval_accuracy = flat_accuracy(logits label_id eval_accuracy + = tmp_eval_accuracy nb_eval_steps + = 1 print("validation accuracy .format(eval_accuracy nb_eval_steps output display information epoch trange wrapper epoch 0%| | 0/4 00:00 < s train loss 0.5381132976395461 epoch 25%| | 1/4 07:54<23:43 474.47s validation accuracy 0.788966049382716 train loss 0.315329696132929 epoch 50%|

| 2/4 15:49<15:49 474.55s validation accuracy 0.836033950617284 train loss 0.1474070605354314 epoch 75%| | 3/4 23:43<07:54 474.53s validation accuracy 0.814429012345679 train loss 0.07655430570461196 epoch 100%|
| 4/4 31:38<00:00 474.58s validation accuracy 0.810570987654321

transformer model evolve quickly deprecation message error occur hug face exception update code accordingly model train display training evaluation loss accuracy value store train_loss_set define beginning program plot measurement @title training evaluation output graph show training process go efficient figure 3.6 training loss batch fine tune bert model model fine tune run prediction predict evaluate holdout dataset bert downstream model train in_domain_train.tsv dataset program prediction holdout test dataset out_of_domain_dev.tsv file goal predict sentence grammatically correct following excerpt code show datum preparation process apply training datum repeat code holdout dataset @title predict evaluate holdout dataset df = pd.read_csv("out_of_domain_dev.tsv delimiter='\t header = names=['sentence_source label label_notes sentence create sentence label list sentence = df.sentence.values need add special token beginning end sentence bert work properly sentence = cls + sentence + sep sentence sentence label = df.label.values tokenized_text = tokenizer.tokenize(sent send sentence program run batch prediction dataloader batch prediction_dataloader add batch gpu batch = tuple(t.to(device t batch unpack input dataloader b_input_id b_input_mask b_label = batch tell model compute store gradient save memory

speed prediction forward pass calculate logit prediction logit = model(b_input_id token_type_id = attention_mask = b logit label prediction move cpu logit label cpu logit = logits['logits'].detach().cpu().numpy label_id = b_labels.to('cpu').numpy prediction true label store store prediction true label program evaluate prediction evaluate matthews correlation coefficient matthews correlation coefficient mcc initially design measure quality binary classification modify multi class correlation coefficient class classification probability prediction tp = true positive tn = true negative fp = false positive fn = false negative brian w. matthews biochemist design 1975 inspire predecessor phi function evolve format follow value produce mcc -1 +1 +1 maximum positive value predic- tion -1 inverse prediction 0

average random prediction glue evaluate linguistic acceptability mcc mcc import sklearn.metric @title evaluate matthew correlation coefficient import evaluate test batch matthew correlation sklearn.metric import matthews_corrcoef set prediction create matthews_set = fine tune bert models mcc value calculate store matthews_set range(len(true_labels matthews = matthews_corrcoef(true_labels[i message library module version change final score base entire test set let look score individual batch sense variability metric batch score individual batch let view score individual batch @title score individual batch output produce mcc value -1 +1 expect mcc value positive good news let evaluation dataset matthews evaluation dataset mcc practical way evaluate classification model program aggregate true value dataset @title matthew evaluation dataset flatten prediction true value aggregate matthew evaluation dataset flat_prediction = item sublist prediction item sublist flat_prediction = np.argmax(flat_predictions axis=1).flatten flat_true_labels = item sublist true_label item sublist mcc produce correlation value 1 +1 0

average prediction -1 inverse 1 perfect case output confirm mcc positive show correlation model dataset final positive evaluation fine tuning bert model overall view bert training framework bert bring bidirectional attention transformer predict sequence left right mask

future token train model limitation mask sequence con- tain meaning look model produce error bert attend token sequence time explore architecture bert use encoder stack transformer bert design step framework step framework pretrain model second step fine tune model build fine tune bert model acceptability judgment downstream task fine tuning process go phase process load dataset load necessary pretraine module model model train performance measure fine tune pretraine model take few machine resource train downstream task scratch fine tune model perform variety task bert prove pretrain model task remarkable produce multitask fine tune model base train parameter bert pretraine model extraordinary fine tune bert models chapter 7 rise suprahuman transformer gpt-3 engine show openai reach zero shot level little fine tuning chapter fine tune bert model chapter chapter 4 pretraine roberta model scratch dig deeply bert framework build pretraine bert like model scratch bert stand bidirectional encoder representations transformers true false bert step framework step 1 pretraine step 2 fine tuning true false fine tune bert model imply training parameter scratch true false bert pretrain downstream task true false bert pretrain masked language modeling mlm true false bert pretrain sentence predictions nsp true false bert pretrain mathematical function true false 8 question answer task downstream task true false 9 bert pretraine model require tokenization true false 10 fine tune bert model take time pretraine true false ashish vaswani noam shazeer niki parmar jakob uszkoreit llion jones aidan n. go- mez lukasz kaiser illia polosukhin 2017 attention need https://arxiv.org/ jacob devlin ming wei chang kenton lee kristina toutanova 2018 bert pretrain- ing

deep bidirectional transformers language understanding https://arxiv.org/ alex warstadt amanpreet singh samuel r. bowman 2018 neural network acceptability corpus linguistic acceptability cola https://nyu-mll.github.io/cola/

documentation hug face model join book discord space join book discord workspace monthly ask session author pretraine roberta model chapter build roberta model scratch model use

brick transformer construction kit need bert model pretrained tokenizer model roberta model build follow step process describe use knowledge transformer acquire previous chapter build model perform language modeling mask token step step chapter 2 getting start architecture transformer model go building block original transformer chapter 3 fine tune bert models fine tune pretrained bert model chapter focus build pretrained transformer model scratch jupyter notebook base hugging face seamless module model name kantaibert kantaibert load compilation immanuel kant book create chapter datum obtain create dataset notebook kantaibert train tokenizer scratch build merge vocabulary file pretraining process kantaibert process dataset initialize trainer train model finally kantaibert use train model perform experimental downstream language modeling task fill mask immanuel kant logic pretraine roberta model scratch end chapter know build transformer model scratch knowledge transformer face industry 4.0 challenge power- ful pretraine transformer gpt-3 engine require development skill implement chapter prepare chapter 7 rise suprahuman transformers gpt-3 engines chapter cover following topic roberta- distilbert like model train tokenizer scratch byte level byte pair encoding save train tokenizer file recreate tokenizer pretraine process initialize roberta model scratch explore configuration model explore 80 million parameter model build dataset trainer initialize trainer pretraine model save model apply model downstream task masked language modeling mlm step describe transformer model go build train tokenizer pretraine transformer chapter train transformer model name kantaibert building block provide hugging face bert like model cover theory building block model chapter 3 fine tune bert models describe kantaibert build knowledge acquire previous chapter kantaibert robustly optimized bert pretraining approach roberta)-like model base architecture bert initial bert model bring innovative feature initial transformer model see chapter 3 roberta increase

innovative feature initial transformer model see chapter 3 roberta increase performance transformer downstream task improve mechanic pretraine process example use wordpiece tokenization go byte level byte pair en- coding bpe method pave way wide variety bert bert

like model chapter kantaibert like bert train masked language modeling mlm mlm language modeling technique mask word sequence transformer model train predict mask word kantaibert train small model 6 layer 12 head 84,095,008 parameter 84 million parameter lot parameter spread 12 head make relatively small model small model pretraine experi- ence smooth step view real time wait hour result kantaibert distilbert like model architecture 6 layer 12 head distilbert distil version bert distilbert suggest contain few parameter roberta model run fast result slightly accurate roberta model know large model achieve excellent performance want run model smartphone miniaturization key technological evolution transformer follow path implementation hugging face approach distil version bert good step forward distillation few parameter method future clever way take good pretraine make efficient need downstream task important possible architecture include run small model smart- phone future transformer ready use api chapter 7 rise suprahuman transformers gpt-3 engines kantaibert implement byte level byte pair encoding tokenizer like gpt-2 special token one roberta bert model use wordpiece token type id

indicate segment token segment separate separation token < /s > kantaibert use custom dataset train tokenizer train transformer model save run mlm example let go build transformer scratch pretraine roberta model scratch build kantaibert scratch build kantaibert 15 step scratch run mlm example open google colaboratory need gmail account upload kantaibert.ipynb github chapter directory title 15 step section similar title notebook cell make easy follow let start load dataset step 1 load dataset ready use dataset provide objective way train compare transformer chapter 5 downstream nlp task transformer explore dataset chapter aim understand training process transformer notebook cell run real time wait hour obtain result choose use work immanuel kant 1724 1804 german philosopher epitome age enlightenment idea introduce human like logic pretrained reasoning downstream reasoning task project gutenberg https://www.gutenberg.org offer wide range free ebook download text format use book want create customize dataset base book compile follow book immanuel

kant text file name kant.txt critique pure reason critique practical reason fundamental principles metaphysic morals kant.txt provide small training dataset train transformer model chapter result obtain remain experimental real life project add complete work immanuel kant rene descartes pascal leibnitz example text file contain raw text book reality vain profess indifference regard inquiry object indifferent humanity dataset download automatically github cell kantaibert.ipynb load kant.txt directory chapter github colab file manager case curl retrieve github @title step 1 load dataset 1.load kant.txt colab file manager 2.downloade file github curl -l https://raw.githubusercontent.com/denis2054/transformers-for-nlp- 2nd edition master chapter04 kant.txt --output kant.txt appear colab file manager pane load download figure 4.1 colab file manager note google colab delete file

restart vm dataset define load program install hugging face transformer step 2 instal hug face transformer need install hugging face transformer tokenizer need tensor- flow instance google colab vm @title step 2 instal hug face transformer will need tensorflow run subsequent cell kant.txt training data prerequisite pretraine roberta model scratch pip uninstall -y tensorflow install transformer master pip install git+https://github.com/huggingface/transformers pip list | grep -e transformers|tokenizer transformer version notebook update 2.9.1 tokenizer version notebook update 0.7.0 output display version instal successfully build transformer transformer version evolve speed version run differ dis- program begin train tokenizer step 3 train tokenizer section program use pretrained tokenizer example pretrained gpt-2 tokenizer training process chapter include train tokenizer hugging face bytelevelbpetokenizer train kant.txt bpe tokenizer break string word substring subword main advantage tokenizer break word minimal component merge small component statistically interesting one example small small small er e tokenizer sm example case word break subword token small unit subword part sm instead simply small chunk string classify unknown unk_token wordpiece level encoding practically disappear model train tokenizer following parameter file = path path dataset vocab_size=52_000 size

tokenizer model length min_frequency=2 minimum frequency threshold special_tokens= list special token case list special token < s > start token < pad > padding token < /s > end token < unk > unknown token < mask > mask token language modeling tokenizer train generate merge substring token analyze frequency let word middle sentence step tokenize string ġthe ġtoken izer string tokenize token ġ whitespace information step replace index table 4.1 index token program run tokenizer expect @title step 3 train tokenizer pathlib import path tokenizer import bytelevelbpetokenizer path = str(x x path(" ").glob("**/*.txt initialize tokenizer tokenizer = bytelevelbpetokenizer customize training pretraine roberta model scratch tokenizer.train(files = path vocab_size=52_000 min_frequency=2 special tokenizer output time take train cpu

path vocab_size=52_000 min_frequency=2 special tokenizer output time take train cpu time

user 14.8 s sys 14.2 s total 29 s wall time 7.72 s tokenizer train ready save step 4 save file disk tokenizer generate file train merges.txt contain merged tokenize substring vocab.json contain index tokenize substring program create kantaibert directory save file @title step 4 save file disk token_dir = /content kantaibert os.path.exists(token_dir program output show file save file appear file manager pane figure 4.2 colab file manager file example small double click view content merge txt contain tokenize substring plan version 0.2 train huggingface tokenizer vocab.json contain index train tokenize dataset file ready process pretraine roberta model scratch step 5 load train tokenizer file load pretraine tokenizer file train tokenizer ready load file @title step 5 load train tokenizer file tokenizers.implementation import bytelevelbpetokenizer tokenizers.processor import bertprocessing tokenizer = bytelevelbpetokenizer tokenizer encode sequence tokenizer.encode("the critique pure reason ") .tokens critique pure reason ġcritique ġof ġpure ġreason ask number token sequence tokenizer.encode("the critique pure reason output 6 token sequence encoding(num_tokens=6 attributes=[id type_id token offset attention_mask special_tokens_mask overflow tokenizer

process token fit bert model variant notebook post processor add start end token example tokenizer._tokenizer.post_processor = bertprocessing let encode post processed sequence tokenizer.encode("the critique pure reason output show 8 token encoding(num_tokens=8 attributes=[id type_id token offset attention_mask special_tokens_mask overflow want add ask tokenizer encode post processed sequence run following cell tokenizer.encode("the critique pure reason ") .tokens output show start end token add bring number token 8 include start end token < s > ġcritique ġof ġpure ġreason < /s >

data training model ready train check system infor- mation machine run notebook step 6 check resource constraint gpu cuda kantaibert run optimal speed graphics processing unit gpu run command nvidia gpu card present @title step 6 check resource constraint gpu nvidia output display information version card figure 4.3 information nvidia card pretraine roberta model scratch output vary google colab vm configuration check sure pytorch see cuda @title check pytorch see cuda result true compute unified device architecture cuda develop nvidia use parallel computing power gpu nvidia gpu cuda appendix ii hardware constraints transformer models ready define configuration model step 7 define configuration model pretraine roberta type transformer model number layer head distilbert transformer model vocabulary size set 52,000 12 attention head 6 layer @title step 7 define configuration model transformer import robertaconfig config = robertaconfig explore configuration detail step 9 initialize model scratch let recreate tokenizer model step 8 reload tokenizer transformer ready load train tokenizer pretrained tokenizer @title step 8 create tokenizer transformers transformer import robertatokenizer tokenizer = robertatokenizer.from_pretrained("./kantaibert max load train tokenizer let initialize roberta model scratch step 9 initialize model scratch section initialize model scratch examine size model program import roberta mask model language modeling @title step 9 initialize model scratch transformer import robertaformaskedlm model initialize configuration define step 7 model = robertaformaskedlm(config = config print model bert model 6 layer 12 head building

block encoder original transformer model present different dimension show excerpt output word_embedding embedding(52000 768 padding_idx=1 position_embedding embedding(514 768 padding_idx=1 token_type_embedding embedding(1 768 layernorm layernorm((768 eps=1e-12 elementwise_affine = true dropout dropout(p=0.1 inplace = false pretraine roberta model scratch query linear(in_features=768 out_features=768 key linear(in_features=768 out_features=768 bias = true value linear(in_features=768 out_features=768 dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=768 layernorm layernorm((768 eps=1e-12

elementwise dropout dropout(p=0.1 inplace = false dense linear(in_features=768 out_features=3072 bias = true dense linear(in_features=3072 out_features=768 bias = true layernorm layernorm((768 eps=1e-12 elementwise dropout dropout(p=0.1 inplace = false time detail output configuration continue know model inside ® -type building block transformer fun analyze example note dropout regularization present sublayer let explore parameter explore parameter model small contain 84,095,008 parameter check size output show approximate number parameter vary trans- version let look parameter store parameter lp calculate length list parameter @title explore parameter output show approximately 108 matrix vector vary transformer model let display 108 matrix vector tensor contain p range(0,lp output display parameter show following excerpt output tensor([[-0.0175 -0.0210 -0.0334 0.0054 -0.0113 0.0183 0.0020 -0.0354 -0.0221 0.0220 -0.0060 -0.0032 0.0001 -0.0002 0.0036 -0.0265 -0.0057 -0.0352 -0.0125 -0.0418 0.0190 -0.0069 0.0175 -0.0308 0.0072 -0.0131 0.0069 0.0002 -0.0234 0.0042 0.0008 0.0281 0.0168 -0.0113 -0.0075 0.0014 minute peek inside parameter understand transformer build pretraine roberta model scratch number parameter calculate take parameter model add example vocabulary 52,000 x dimension 768 size vector 1 x 768 dimension find note dmodel = 768 12 head model dimension dk head = = 64 show optimize lego ® concept building block transformer number parameter model calculate figure 84,095,008 reach hover lp notebook shape torch tensor figure 4.4 lp note number vary

depend version transformer module use count number parameter tensor program initialize parameter counter name np number parameter go lp 108 number element list parameter @title count parameter p range(0,lp):#number tensor parameter matrix vector different size example 768 x 768 768 x 1 parameter dimensional dimensional l2 = len(lp[p][0 check 2d l2=1 2d 1d parameter dimension second dimension l2>0 pl2 = true 2 dimension = true parameter dimension second dimension l2=1 pl2 = false 2 dimension = false l1

size dimension parameter l3 size parameter define add parameter step loop np+=l3 number parameter tensor obtain sum parameter want exactly number parameter transformer model calculate print(p l1,l2,l3 display size parameter print(p l1,l3 display size parameter print(np total number parameter note parameter dimension pl2 = false display di- output list number parameter calculate tensor model show follow excerpt 0 52000 768 39936000 1 514 768 394752 2 1 768 768 pretraine roberta model scratch 3 768 768 4 768 768 5 768 768 589824 6 768 768 7 768 768 589824 8 768 768 9 768 768 589824 10 768 768 total number parameter roberta model display end list number parameter vary version library know precisely number parameter represent transformer model minute look output configuration content parame- ter size parameter point precise mental representation building block model program build dataset step 10 build dataset program load dataset line line generate sample batch training block_size=128 limit length example @title step 10 build dataset transformer import linebylinetextdataset dataset = linebylinetextdataset output show hugging face invest considerable resource opti- mize time take process datum cpu time user 8.48 s sys 234 m total 8.71 s wall time 3.88 s wall time actual time processor active optimize program define data collator create object backpropagation step 11 define data collator need run data collator initialize trainer data collator sample dataset collate batch result dictionary like object prepare batch sample process mlm set mlm = true set number mask token train mlm_probability=0.15 determine percentage token mask pretraine process initialize data_collator tokenizer mlm activate proportion mask token set 0.15

@title step 11 define data collator transformer import datacollatorforlanguagemodeling data_collator = datacollatorforlanguagemodeling tokenizer = tokenizer mlm = true mlm_probability=0.15 ready initialize trainer step 12 initialize trainer previous step prepare information require initialize trainer dataset tokenize load model build data collator create program initialize trainer educational purpose program train model quickly number epoch limit gpu come handy share batch multi process training task @title step 12 initialize trainer transformer import trainer trainingargument training_args = trainingargument pretraine roberta model scratch trainer = trainer model ready training step 13 pretraine model ready trainer launch line code @title step 13 pre training model output display training process real time show loss learning rate epoch 1/1 17:59<00:00 1079.91s 2672/2672 17:59<00:00 2.47it s loss 5.6455852394104005 learning_rate 4.06437125748503e-05 epoch 0.18712574850299402 step 500 loss 4.940259679794312 learning_rate 3.12874251497006e-05 epoch 0.37425149700598803 step 1000 loss 4.639936000347137 learning_rate 2.1931137724550898e-05 epoch 0.561377245508982 step 1500 loss 4.361462069988251 learning_rate 1.2574850299401197e-05 epoch 0.7485029940119761 step 2000 loss 4.228510192394257 learning_rate 3.218562874251497e-06 epoch 0.9356287425149701 step 2500 cpu time user 11min 36s sys 6min 25s total 18min 2s wall time 17min 59s model train time save work step 14 save final model + tokenizer + config disk save model configuration @title step 14 save final model(+tokenizer + config disk click refresh file manager file appear figure 4.5 colab file manager config.json pytorh_model.bin training_args.bin appear file manager merges.txt vocab.json contain pretraine tokenization dataset build model scratch let import pipeline perform language modeling task pretraine model tokenizer step 15 language modeling fillmaskpipeline import language modeling fill mask task use train model train tokenizer perform mlm @title step 15 language modeling fillmaskpipeline transformer import pipeline fill_mask = pipeline pretraine roberta model scratch ask model think like immanuel kant fill_mask("human thinking involve human < mask > output likely change run pretraine

model scratch limited datum output obtain run

interesting introduce conceptual language modeling sequence < s > human thinking involve human reason.</s > sequence < s > human thinking involve human object.</s > sequence < s > human thinking involve human priori.</s > sequence < s > human thinking involve human conception.</s > sequence < s > human thinking involve human experience.</s >

prediction vary run time hugging face update model follow output come human thinking involve human reason goal train transformer model interesting human like prediction result experimental subject variation training process change time train model model require datum age enlightenment thinker goal model create dataset train transformer specific type complex language modeling task thank transformer beginning new era ai train transformer scratch time imagine personal corporate environment create dataset specific task train scratch use area interest company project experiment fascinating world transformer construction kit model like share hugging face community model appear hugging face model page https://huggingface.co/models upload model step instruction describe page https:// download model hugging face community share new idea personal professional project chapter build kantaibert roberta like model transformer scratch building block provide hugging face start load customize dataset specific topic relate work im- manuel kant load exist dataset create depend goal see customize dataset provide insight way transformer model think experimental approach limit large dataset train model educational purpose kantaibert project train tokenizer kant.txt dataset train merge txt vocab.json file save tokenizer recreate pretraine file kantai- bert build customize dataset define data collator process training batch backpropagation trainer initialize explore parameter roberta model detail model train save finally save model load downstream language modeling task goal fill mask immanuel kant logic pretraine roberta model scratch door wide open experiment exist customize dataset result share model hugging face community transformer data drive use advantage discover

new way transformer ready learn run ready use transformer engine api require pretraining fine tuning chapter 7 rise suprahuman transformers gpt-3 engines future ai knowledge chapter past chapter ready chapter downstream nlp tasks transformers continue preparation implement transformer roberta use byte level byte pair encoding tokenizer true false 2 train hugging face tokenizer produce merges.txt vocab.json true false roberta use token type id

produce merges.txt vocab.json true false roberta use token type id true false distilbert 6 layer 12 head true false 5 transformer model 80 million parameter enormous true false train tokenizer true false bert like model 6 decoder layer true false 8

masked language modeling mlm predict word contain mask token 9 bert like model self attention sublayer true false 10 data collator helpful backpropagation true false yinhan liu myle ott naman goyal jingfei du mandar joshi danqi chen omer levy mike lewis luke zettlemoyer veselin stoyano 2019 roberta robustly optimize bert pretraine approach https://arxiv.org/abs/1907.11692 hugging face tokenizer documentation https://huggingface.co/transformers/ hugging face reference notebook https://colab.research.google.com/github/ hugging face reference blog https://colab.research.google.com/github/ bert https://huggingface.co/transformers/model_doc/bert.html distilbert https://arxiv.org/pdf/1910.01108.pdf roberta https://huggingface.co/transformers/model_doc/roberta.html distilbert https://huggingface.co/transformers/model_doc/ join book discord space join book discord workspace monthly ask session author downstream nlp task transformers reveal potential unleash pretrained model watch perform downstream natural language understanding nlu task take lot time effort pretrain fine tune transformer model effort worthwhile multi million parameter transformer model action range nlu task begin chapter quest outperform human baseline human baseline represent performance human nlu task human learn transduction

early age quickly develop inductive thinking

human perceive world directly sense machine intelligence rely entirely perception transcribe word sense language measure performance transformer measure natural language processing nlp task remain straightforward approach involve accuracy score form base true false result result obtain benchmark task dataset superglue example wonderful example google deepmind facebook ai university new york university washington work set high standard measure nlp performance finally explore downstream task standard sentiment treebank sst-2 linguistic acceptability winograd schema downstream nlp task transformers transformers rapidly take nlp level outperform model de- sign benchmark task alternative transformer architecture continue emerge evolve chapter cover following topic machine versus human intelligence transduction induction nlp transduction induction process measure transformer performance versus human baseline measurement method accuracy f1 score mcc benchmark task dataset superglue downstream task linguistic acceptability cola sentiment analysis sst-2 let start understand human machine represent language transduction inductive inheritance emergence automated machine learning automl mean api automate cloud ai platform deeply change job description ai specialist google vertex example boast reduction 80 development require implement ml suggest anybody implement ml ready use system mean 80 reduction workforce developer think industry 4.0 ai specialist assemble ai add value project transformer possess unique ability apply knowledge task learn bert transformer example acquire language sequence sequence mask language modeling bert transformer fine tune perform downstream task learn scratch industry 4.0 nlp ai specialist invest source code knowledge ai guru team section mind experiment use graph transformer represent human machine sense information language machine sense information different way human reach efficient result figure 5.1 mind experiment design transformer architecture layer sublayer show deceptive similarity human machine let study learning process trans- model understand downstream task figure 5.1 human ml method example n=2

conceptual representation layer layer human build accumulate knowledge generation generation machine process machine use output input human intelligence stack left figure 5.1 input human perception raw event layer 0 output language perceive event sense child gradually output burble language structure language downstream nlp task transformer human transduction go trial error process transduction mean structure perceive represent pattern example represen- tation world apply inductive thinking inductive thinking rely quality transduction example child force nap early afternoon famous child psychologist piaget find lead child say example take nap afternoon child see event create link transduction make inference generalize induction human notice pattern transduction generalize induc- tion train trial error understand event relate trained_relate event = sunrise light sunset dark dark cloud rain blue sky running food good fire warm snow cold time train understand million relate event new generation human start scratch fine tune task previous generation teach fire burn example child know knowledge fine tune form fire candle wildfire volcano instance fire finally human transcribe know imagine predict write language output layer 0 bear human input layer layer 1 vast train fine tune knowledge human perceive massive amount event transduction induction training fine tuning sublayer previous transcribe event stem odor emotion situation experience make human unique machine access individual identity human personal perception word homogeneous way specific individual machine take masse heterogeneous unfiltered impersonal datum goal machine perform impersonal efficient task goal human infinite approach loop go layer 0 layer 1 layer 0 raw result fascinating need learn train native language scratch acquire summarization ability use pretrained knowledge adjust fine tune transformer process different way machine intelligence stack machine learn basic task like one describe chapter perform hundred task sequence learn predict right figure 5.1 input machine second hand information form language output input machine analyze language point human machine history

language output input machine analyze language point human machine history computer vision identify image contain grammatical structure language speech recognition convert sound word bring write language music pattern recognition lead objective concept express word machine start handicap impose artificial disadvantage machine rely random quality language output perform transduction connect token subword occur lan- build induction transduction train induction base token produce pattern token let stop point peek process attention sublayer work hard produce valid induction transformer model exclude recurrence base learning operation self attention heighten vision model attention sublayer advantage human point process mil- lion example inductive thinking operation like find pattern sequence transduction induction memorize pattern parameter store model acquire language understanding ability substantial data volume excellent nlp transformer algorithm computer power thank deep understanding language ready run hundred task train downstream nlp task transformers transformer like human acquire language understanding limited number task like detect connection transduction generalize induc- transformer model reach fine tune sub layer machine intelligence react like start train scratch perform new task like consider downstream task require fine tuning need learn answer question start learn language scratch transformer model fine tune param- eter like section see transformer model struggle learn way start handicap moment rely perception transcribe language access infinitely datum massive computing power let measure transformer performance versus human baselines transformer performance versus human baselines transformer like human fine tune perform downstream task inherit property pretrained model pretrained model provide architecture language representation parameter pretrained model train key task acquire general knowledge language fine- tune model train downstream task transformer model use task pretraining potentially task pretraine fine tune nlp model need evaluate standard method section key measurement method main benchmark task dataset let start go key metric method evaluate model metric impossible compare transformer model transformer model nlp model universal measurement system

compare transformer model transformer model nlp model universal measurement system use metric section analyze measurement scoring method glue accuracy score variant use practical evaluation score function calcu- late straightforward true false value result model output match correct prediction give subset samplesi set sample basic function obtain
1 result subset correct 0 false let examine flexible f1 score f1 score introduce flexible approach help face dataset con- taine uneven class distribution f1 score use weight value precision recall weight average precision recall value f1score= 2 precision recall)/(precision + recall equation true t positive p false f positive p false f negative n plug precision p recall r equation f1 score view harmonic mean reciprocal arithmetic mean precision p recall r let review mcc approach downstream nlp task transformers matthews correlation coefficient mcc mcc describe implement evaluate matthews correlation coefficient section chapter 3 fine tune bert models mcc compute measurement true positive tp true negative tn false positive fp false negative fn mcc summarize following equation mcc provide excellent metric binary classification model size class good idea measure give transformer model result compare transformer model nlp model measurement scoring method mind let look benchmark task dataset benchmark task dataset prerequisite require prove transformer reach state art per- dataset drive task metric describe evaluate model metric section chapter begin explore superglue benchmark illustrate evaluation process glue superglue superglue benchmark design public wang et al 2019 wang et al 2019 design general language understanding evaluation glue benchmark motivation glue benchmark useful nlu applicable wide range task relatively small glue dataset design encourage nlu model solve set task performance nlu model boost arrival transformer begin exceed level average human glue leaderboard december 2021 glue leaderboard available https://gluebenchmark.com/leaderboard show remarkable display nlu talent retain rnn cnn idea mainly focus ground break transformer model follow

rnn cnn idea mainly focus ground break transformer model follow excerpt leaderboard show

leader position glue hu- figure 5.2 glue leaderboard december 2021 new model human baselines ranking constantly change ranking idea far classical nlp transformer take notice glue human baselines position show nlu model surpass non expert human glue task human baselines represent human achieve ai outperform human december 2021 human baselines position 17 problem standard beat challenging fish benchmark dataset improve model blindly downstream nlp task transformer notice transformer model

take lead like think glue superglue point word chaos order language understanding understanding glue make word fit language glue leaderboard continuously evolve nlu progress wang et al 2019 introduce superglue set high standard human baselines introduce high human baselines standard wang et al 2019 recognize limit glue design superglue difficult superglue immediately establish human baselines rank 1 december 2020 show follow excerpt leaderboard https://super.gluebenchmark.com/leaderboard figure 5.3 superglue leaderboard 2.0 december 2020 superglue leaderboard evolve produce well nlu model 2021 trans- former surpass human baselines december 2021 human baselines go rank 5 show figure 5.4 figure 5.4 superglue leaderboard 2.0 december 2021 ai algorithm ranking constantly change new innovative model arrive ranking idea hard battle nlp supremacy fight let evaluation process work superglue evaluation process wang et al 2019 select task superglue benchmark selection criterion task strict glue example task understand text reason level reasoning human expert level performance sufficient replace human task downstream nlp task transformer superglue task present ready use list figure 5.5 superglue task task list interactive https://super.gluebenchmark.com/tasks task contain link require information perform task downstream task fine tune pretrained model identifier abbreviation short version download download link dataset info offer great detail link paper website team design dataset drive task(s metric measurement score evaluate model superglue provide task instruction software dataset paper website describe problem solve team run benchmark task reach leaderboard result display figure 5.6 superglue task score superglue display overall score score task

example let instruction wang et al 2019 provide choice plausible answers copa task table 6 paper step read remarkable paper write roemmele et al 2011 nutshell goal nlu model demonstrate machine thinking human thinking course potential case transformer choose plausible answer question dataset provide premise transformer model find plausible answer premise knock neighbor door happen result alternative 1 neighbor invite alternative 2

neighbor door happen result alternative 1 neighbor invite alternative 2 neighbor leave house question require second human answer show require commonsense machine thinking copa.zip ready use dataset download directly superglue task page metric provide make process equal reliable participant benchmark race downstream nlp task transformer example difficult transformer near copa

human baselines figure 5.7 rank 5 task account figure 5.7 superglue result copa incredible transformer climb leaderboard ladder short time beginning new idea emerge nearly month introduce copa let define seven superglue benchmark task define superglue benchmark task task pretraine task generate train model task downstream task model fine tune goal superglue give nlu model perform multiple downstream task fine tuning multi task model one prove thinking power transformer power transformer reside ability perform multiple task pretraine model apply fine tune downstream task original transformer model variant lead glue superglue task continue focus superglue downstream task human baselines tough beat previous section go copa section seven task define wang et al 2019 table 2 paper let continue boolean question task boolq boolean yes answer task dataset define superglue contain 15,942 naturally occur example raw sample line 3 train.jsonl dataset contain passage question answer true question windows movie maker windows essential passage windows movie maker windows movie maker know windows live movie maker windows 7 discontinue video editing software microsoft windows essentials software suite offer ability create edit video publish onedrive facebook vimeo youtube flickr idx 2 dataset provide change time concept

remain let examine cb task require human machine focus commitment bank cb commitment bank cb difficult entailment task ask transformer model read premise examine hypothesis build premise example hypothesis confirm premise contradict transformer model label hypothesis neutral entailment contradiction premise example dataset contain discourse natural discourse following sample 77 take train.jsonl training dataset show difficult cb task premise susweca mean dragonfly sioux know tell paul meet hypothesis susweca paul meet label entailment idx 77 look multi sentence problem multi sentence reading comprehension multirc multi sentence reading comprehension multirc ask model read text choose possible choice task difficult human machine model present text question possible answer question 0

false 1 true label downstream nlp task transformer let second sample train.jsonl text text rally take place october 17 shooting february 29 standard filmmaking technique interpret smooth distortion \"moore work deprive context guide mind fill vacuum completely false idea brilliantly unethically done.\ note \"from cold dead hands\ simply moore way introduce heston moore critic view certainly \"attribute speech uttered\ note deceiver critic claim concern georgetown hoya interview heston ask rolland write \"there indication heston recognize kayla rolland case.\ naive extreme heston president nra keep date prominent case gun violence respond interview certainly know case point nra website excerpt case highlighting phrase \"48 hour kayla rolland pronounce dead\ valid criticism far deliberate distortion example fact easy miss moore fast pace editing reason sentence highlight deceive viewer believe heston hurry flint immediately hold rally obvious simply highlight mention \"kayla rolland\ text sample contain question illustrate task investigate model predict correct label notice information model ask obtain distribute text question kayla rolland shoot text february 17 idx 168 label 0 text february 29 idx 169 label 1 text october 29 idx 170 label 0 text october 17 idx 171 label 0 text february 17 idx 172 label 0 idx 26 question president nra february 29 answers text charleton heston idx 173 label 1 text moore idx 174 label 0 text george hoya idx 175 label 0 text rolland idx 176 label 0 text hoya idx 177 label 0 text kayla idx 178 label 0 idx 27 point

admire performance single fine tune pretrained model difficult downstream task let reading comprehension task reading comprehension commonsense reasoning dataset reading comprehension commonsense reasoning dataset record represent challenging task dataset contain 120,000 query 70,000 news article transformer use common sense reasoning solve problem let examine sample train.jsonl source daily mail passage contain text indication entity passage begin text text peruvian tribe revere inca fierce hunting skill formidable warrior cling traditional existence coca grow valley south america share land drug trafficker rebel illegal logger ashaninka indians large group indigenous people mountainous nation amazon region settlement sparse cent peru 30 million population battle rival tribe territory food native rule rainforest south america ashaninka rarely know peace.\n@highlight\nthe ashaninka tribe share amazon like incas hundred year ago\n@highlight\ nthey force share land year conflict force rebel drug dealer forest\n@highlight\n despite settle valley rich valuable coca live poor pre industrial downstream nlp tasks transformer entity indicate show following excerpt entity start 2,"end 9 start 711,"end 715 finally model answer query find proper value placeholder query innocence youth @placeholder young generation turn back tribal life move city living condition well transformer model go problem face entailment task recognize textual entailment rte recognize textual entailment rte transformer model read premise ex- amine hypothesis predict label entailment hypothesis status let examine sample 19 train.jsonl dataset premise u.s. crude settle $ 1.32 lower $ 42.83 barrel hypothesis crude light american lower closing 1.32 dollar 42.83 dollar barrel label not_entailment > idx rte require understanding logic let word context task words context wic words context wic follow winograd task test model ability process ambiguous word wic multi task transformer analyze sentence determine target word meaning sentence let examine sample train.jsonl dataset target word specify model read sentence contain target word sentence1 want come place later sentence2 political system place prominent train.jsonl

specify sample index value label position target word sentence1(start1 end1 sentence2(start2 end2 daunting task transformer model face winograd task winograd schema challenge wsc winograd schema task name terry winograd transformer train able solve disambiguation problem dataset contain sentence target slight difference gender pronoun constitute coreference resolution problem challenging task perform transformer architecture allow self attention ideal task sentence contain occupation participant pronoun problem solve find pronoun coreferent occupation participant let examine sample take train.jsonl sample ask model read text text > pour water bottle cup wsc ask model find target pronoun token number 10 start target span2_index 10 ask model determine refer cup span1_text cup sample index 4 label true idx 4 label true downstream nlp task transformer go main superglue task task understand architecture transformer mechanism bench- mark task rapidly adapt model benchmark let run downstream task run downstream task section jump transformer car drive bit model task run section understand process run task quickly understand human baseline task downstream task fine tune transformer task inherit model parameter pretrained transformer model downstream task perspective pretrained model run fine tune task mean depend model task downstream fully pretrain model section consider task downstream pretrain model evolve database benchmark method accuracy measurement method leaderboard criterion structure human thought reflect downstream task chapter remain let start cola corpus linguistic acceptability cola corpus linguistic acceptability cola glue task https://gluebenchmark.com/tasks contain thousand sample english sentence annotate grammatical acceptability goal alex warstadt et al 2019 evaluate linguistic competence nlp mod- el judge linguistic acceptability sentence nlp model expect classify sentence label grammatical ungrammatical sentence label 0 sen- tence grammatically acceptable sentence label 1 sentence grammatically acceptable example classification = 1 yell hoarse classification = 0

yell bert_fine_tuning_sentence_classification_gpu.ipynb chapter 3 fine tune bert models view

bert model fine tune cola dataset cola datum @title load dataset source dataset https://nyu-mll.github.io/cola/ df = pd.read_csv("in_domain_train.tsv delimiter='\t header = names=['sentence_source label label_notes sentence load pretrained bert model @title load hug face bert uncased base model model = bertforsequenceclassification.from_pretrained("bert base uncased finally measurement method metric mcc describe eval- uating matthews correlation coefficient section chapter 3 fine tuning bert models early chapter refer section mathematical description mcc time rerun source code necessary sentence grammatically unacceptable convey sentiment sentiment analysis add form empathy machine stanford sentiment treebank sst-2 stanford sentiment treebank sst-2 contain movie review section describe sst-2 binary classification task dataset possible classify sentiment range 0 negative n positive socher et al 2013 take sentiment analysis binary positive negative nlp classifica- tion explore sst-2 multi label sentiment classification transformer model chapter 12 detect customer emotion prediction section run sample take sst hugging face transformer pipeline model illustrate binary classification downstream nlp tasks transformers open transformer_tasks.ipynb run following cell contain positive negative movie review take sst @title sst-2 binary classification transformer import pipeline nlp = pipeline("sentiment analysis print(nlp("if like movie fun wasabi good place start "),"if like movie fun wasabi good place start print(nlp("effective tepid biopic "),"effective tepid output accurate label positive score 0.999825656414032 like movie fun wasabi good place start label negative score 0.9974064230918884 effective too- sst-2 task evaluate accuracy metric classify sentiment sequence let sentence sequence paraphrase microsoft research paraphrase corpus mrpc microsoft research paraphrase corpus mrpc glue task contain pair sentence extract new source web pair annotate human indicate sentence equivalent base closely relate property let run sample hugging face bert model open transformer_tasks.ipynb following cell run sample take mrpc @title sequence classification paraphrase classification transformer import autotokenizer import tensorflow

tf tokenizer = autotokenizer.from_pretrained("bert base case finetuned- model = tfautomodelforsequenceclassification.from_pretrained("bert base- class = paraphrase paraphrase sequence_a = dvd cca appeal state supreme court sequence_b = dvd cca appeal decision u.s. supreme paraphrase = tokenizer.encode_plus(sequence_a sequence_b return paraphrase_classification_logits = model(paraphrase)[0 paraphrase_results = tf.nn.softmax(paraphrase_classification_logits print(sequence_b paraphrase range(len(classes print(f"{classes[i round(paraphrase_results[i 100)}% output accurate message warn model need dvd cca appeal decision u.s. supreme court paraphrase 8.0 paraphrase 92.0 mrpc task measure f1 accuracy score method let run winograd schema describe winograd schema chapter winograd schema challenge wsc section training set english happen ask transformer model solve pronoun gender problem en- glish french translation french different spelling noun grammatical gender follow sentence contain pronoun refer word car garage transformer disambiguate pronoun downstream nlp task transformers open transformer_tasks.ipynb winograd cell run example transformers import pipeline translator = pipeline("translation_en_to_fr print(translator("the car garage translation

perfect translation_text la voiture ne pouvait pas aller dans le garage parce qu'elle était trop grosse transformer detect word refer word car feminine form feminine form apply adjective big elle mean french translation masculine form il mean grosse feminine form translation word big masculine form gros give transformer difficult winograd schema solve produce right answer dataset drive nlu task available explore through- book add building block toolbox transformer chapter analyze difference human language representation process way machine intelligence perform transduction see transformer rely output incredibly complex thought process express write language language remain precise way express massive information machine sense convert speech text extract meaning raw dataset explore measure performance multi task transformer transformer ability obtain rank result downstream task unique nlp history go tough superglue task bring transformer rank glue superglue

leaderboard boolq cb wic task cover means easy process human go example downstream task difficulty transformer model face prove efficiency transformer prove value outperform nlu architecture illus- trate simple implement downstream fine tune task run task google colaboratory notebook hugging face pipeline transformer winograd schema give transformer difficult task solve winograd disambig- uation problem english french translation chapter chapter 6 machine translation transformer translation task step build translation model trax machine intelligence use datum human prediction true false superglue difficult glue nlp model true false boolq expect binary answer true false wic stand word context true false recognize textual entailment rte detect sequence entail 6

winograd schema predict verb spell correctly true false transformer model occupy rank glue superglue true false 8 human baselines standard define tough attain superglue true false transformer model beat superglue human baselines standard true false 10 variant transformer model outperform rnn cnn model true false alex wang yada pruksachatkun nikita nangia amanpreet singh julian michael felix hill omer levy samuel r. bowman 2019 superglue sticki benchmark general pur- pose language understanding system https://w4ngatang.github.io/static/papers/ downstream nlp task transformer alex wang yada pruksachatkun nikita nangia amanpreet singh julian michael felix hill omer levy samuel r. bowman 2019 glue multi task benchmark analysis platform natural language understanding yu sun shuohuan wang yukun li shikun feng hao tian hua wu haifeng wang 2019 ernie 2.0 continual pretraining framework language understanding https://arxiv melissa roemmele cosmin adrian bejan andrew s. gordon 2011 choice plausible alternatives evaluation commonsense causal reasoning https://people.ict.usc richard socher alex perelygin jean y. wu jason chuang christopher d. manning andrew y. ng christopher potts 2013 recursive deep models semantic compositionality sentiment treebank https://nlp.stanford.edu/~socherr/emnlp2013_rntn.pdf thomas wolf lysandre debut victor sanh julien chaumond clement delangue anthony moi pierric cistac tim rault rémi louf morgan

funtowicz jamie brew 2019 huggingface trans- former state art natural language processing https://arxiv.org/abs/1910.03771 hugging face transformer usage https://huggingface.co/transformers/usage.html

join book discord space join book discord workspace monthly ask session author machine translation humans master sequence transduction transfer representation object easily imagine mental representation sequence somebody say flower garden beautiful easily visualize garden flower image garden see garden imagine chirp bird scent flower machine learn transduction scratch numerical representation recurrent convolutional approach produce interesting result reach significant bleu translation evaluation score translate require representation language transpose language b. transformer model self attention innovation increase analytic ability machine in- telligence sequence language adequately represent attempt translate language b. self attention bring level intelligence require machine obtain well bleu score seminal attention need transformer obtain good result english german english french translation 2017 score improve point book cover essential aspect transformer architecture transformer train roberta model scratch fine tune bert evaluate fine- tune bert explore downstream task transformer example machine translation transformer chapter machine translation additional topic define machine translation preprocess workshop machine translation wmt dataset finally implement machine translation chapter cover following topic define machine translation human transduction translation machine transduction translation preprocesse wmt dataset evaluate machine translation bleu introduce google translate api initialize english german problem trax step define machine translation define machine translation vaswani et al 2017 tackle difficult nlp problem design trans- human baseline machine translation reach human machine intelligence designer stop vaswani et al 2017 publish transformer architecture achieve state art bleu result section define machine translation machine translation process repro- duce human translation machine transduction output figure 6.1 machine translation process general idea figure 6.1 machine

following step choose sentence translate learn word relate hundred million parameter learn way word refer use machine transduction transfer learn parameter new sequence choose candidate translation word sequence process start sentence translate source language a. process end output contain translate sentence language b. intermediate calculation human transduction translation human

translate sentence language b. intermediate calculation human transduction translation human interpreter european parliament instance translate sentence word word word word translation sense lack proper grammat- ical structure produce right translation context word ignore human transduction take sentence language build cognitive representation sentence meaning interpreter oral translation translator write translation european parliament transform transduction interpretation sentence language b. translation interpreter translator language b reference sentence notice reference machine translation process describe figure 6.1 human translator translate sentence sentence b time real life translator translate sentence real life example find french english translation les essais montaigne sentence original french version find version sentence b note reference 1 n. european parliament day notice interpreter trans- late limit time hour example interpreter take interpreter style like writer different style sentence source language repeat person time day translate reference sentence b version reference = reference 1 reference 2, reference n machine translation transformer machines find way think way human translator machine transduction translation transduction process original transformer architecture use encoder stack decoder stack model parameter represent reference sequence refer output sequence reference output prediction problem single output prediction transformer like human produce result refer change train differently use different transformer model immediately realize human baseline human transduction representation language sequence challenge progress evaluation machine translation prove nlp progress determine solution well nlp challenger lab organization refer dataset comparison valid let explore wmt dataset preprocesse wmt dataset vaswani et al 2017 present transformer

achievement wmt 2014 english ger- man translation task wmt 2014 english french translation task transformer achieve state art bleu score bleu describe evaluate machine trans- lation bleu section chapter 2014 wmt contain european language dataset dataset contain datum take version 7 europarl corpus french english dataset european parliament proceedings parallel corpus 1996 2011 https://www.statmt.org/ download file extract preprocess parallel load clear reduce size

https://www.statmt.org/ download file extract preprocess parallel load clear reduce size corpus let start preprocessing preprocesse raw datum section preprocess europarl-v7.fr-en.en europarl-v7.fr-en.fr open read.py chapter github directory ensure europarl file directory read.py program begin standard python function pickle dump serialize output pickle import dump define function load file memory load doc memory open file read file = open(filename mode='rt encoding='utf-8 read text text = file.read close file load document split sentence split load document sentence short long length retrieve short long sentence length length = len(s.split s sentence

return min(lengths max(length import sentence line clean avoid train useless noisy token line normalize tokenize white space convert lowercase punctuation remove token non printable character remove token contain number exclude clean line store string machine translation transformer program run cleaning function return clean append string clean line clean = list prepare regex char filtering re_print = re.compile('[^%s re.escape(string.printable prepare translation table remove punctuation table = str.maketrans string.punctuation line line normalize unicode character line = unicodedata.normalize('nfd line).encode('ascii line = line.decode('utf-8 tokenize white space line = line.split convert low case line = word.lower word line remove punctuation token line = word.translate(table word line remove non printable char form token line = re_print.sub w w line remove token number line = word word line word.isalpha store string define key function prepare dataset english datum load clean load

english datum filename = europarl-v7.fr-en.en doc = load_doc(filename sentence = to_sentences(doc minlen maxlen = sentence_lengths(sentence print('english datum sentences=%d min=%d max=%d len(sentence dataset clean pickle dump serialize file name english.pkl filename = english.pkl outfile = open(filename,'wb output show key statistic confirm english.pkl save english datum sentences=2007723 min=0 max=668 repeat process french datum dump serialize file name load french datum filename = europarl-v7.fr-en.fr doc = load_doc(filename sentence = to_sentences(doc minlen maxlen = sentence_lengths(sentence print('french datum sentences=%d min=%d max=%d len(sentence filename = french.pkl outfile = open(filename,'wb output show key statistic french dataset confirm french.pkl save french datum sentences=2007723 min=0 max=693

main preprocessing need sure dataset contain noisy confusing token machine translation transformer finalize preprocessing dataset open read_clean.py directory read.py process define func- tion load dataset clean previous section save preprocessing finalize pickle import load pickle import dump collection import counter load clean dataset return load(open(filename rb save list clean sentence file def save_clean_sentences(sentence filename dump(sentence open(filename wb print('save s filename define function create vocabulary counter important know time word sequence parse example word dataset contain million line waste energy precious gpu resource learn let define counter create frequency table word vocab = counter line line token = line.split vocabulary counter detect word frequency min_occurrence remove word frequency threshold def trim_vocab(vocab min_occurrence token = k k c vocab.items c > = min_occurrence case min_occurrence=5 word equal threshold remove avoid waste training model time analyze deal vocabulary oov word oov word misspell word abbreviation word fit standard vocabulary representation use automatic spelling solve problem example simply replace oov word unk unknown token mark oov unk line def update_dataset(lines vocab new_line = list line line new_token = list token line.split token vocab new_line = .join(new_tokens run function english dataset save output

display 20 line load english dataset filename = english.pkl line = load_clean_sentences(filename calculate vocabulary vocab = to_vocab(lines print('english vocabulary d len(vocab reduce vocabulary vocab = trim_vocab(vocab 5 print('new english vocabulary d len(vocab mark vocabulary word line = update_dataset(lines vocab save update dataset filename = english_vocab.pkl machine translation transformer spot check range(20 output function vocabulary compression obtain english vocabulary 105357 new english vocabulary 41746 preprocesse dataset save output function display 20 line show line 0 resumption session line 1 declare resume session european parliament adjourn friday december like wish happy new year hope enjoy pleasant festive period line 2 see dread millennium bug fail materialise people number country suffer series natural disaster truly dreadful line 3 request debate subject course day partsession let run function french dataset save output display 20 line load french dataset filename = french.pkl line = load_clean_sentences(filename calculate vocabulary vocab = to_vocab(lines

print('french vocabulary d len(vocab reduce vocabulary vocab = trim_vocab(vocab 5 print('new french vocabulary d len(vocab mark vocabulary word line = update_dataset(lines vocab save update dataset filename = french_vocab.pkl spot check range(20 output function vocabulary compression obtain french vocabulary 141642 new french vocabulary 58800 preprocesse dataset save output function display 20 line show line 0 reprise de la session line 1 je declare reprise la session du parlement europeen qui avait ete interrompue le vendredi decembre dernier et je vous renouvelle tous mes vux en esperant que vous avez passe de bonnes vacances line 2 comme vous avez pu le constater le grand bogue de lan ne sest pas produit en revanche les citoyens dun certain nombre de nos pays ont ete victimes de catastrophes naturelles qui ont vraiment ete terribles line 3 vous avez souhaite un debat ce sujet dans les prochains jours au cours de cette periode de session section show raw datum process training dataset ready plug transformer train line french dataset sentence translate line english

dataset reference machine translation model machine translation model produce english candidate translation match reference bleu provide method evaluate candidate translation produce machine translation evaluate machine translation bleu papineni et al 2002 come efficient way evaluate human translation human baseline difficult define realize obtain efficient result compare human translation machine translation word word papineni et al 2002 name method bilingual evaluation understudy score bleu machine translation transformer section use natural language toolkit nltk implement bleu begin geometric evaluation bleu method compare part candidate sentence reference sentence open bleu.py chapter directory

github repository book program import nltk module nltk.translate.bleu_score import sentence_bleu nltk.translate.bleu_score import smoothingfunction simulate comparison candidate translation produce machine trans- lation model actual translation(s reference dataset remember sentence repeat time translate different translator different way make challenging find efficient evaluation strategy program evaluate reference reference = cat like milk cat like milk candidate = cat like milk score = sentence_bleu(reference candidate print('example 1 score reference = cat like milk candidate = cat likes milk score = sentence_bleu(reference candidate print('example 2 score score example 1 example 1 1.0 example 2 1.0 straightforward evaluation p candidate c reference r number correct token find c n represent geometric function geometric approach rigid look 3 gram overlap example reference = cat likes milk candidate = cat enjoys','milk score = sentence_bleu(reference candidate print('example 3 score output severe look 3 gram overlap warning warning module nltk\translate\bleu_score.py line 490 corpus sentence contain 0 count 3 gram overlap bleu score undesirable use smoothingfunction example 3 0.7071067811865475

human score 1 0.7 hyperparameter change approach remain rigid warning code good announce section message vary version program run stochastic papineni et al 2002 come modify unigram approach idea count word occurrence reference sentence ensure word

evaluate machine translation transformer consider following example explain papineni et al 2002 reference 1 cat mat reference 2 cat mat consider following candidate sequence candidate look number word candidate sentence 7 occurrence word present reference 1 sentence 2 occurrence word standard unigram precision 7/7 modify unigram precision 2/7 note bleu function output warning agree suggest smoothing let add smoothing technique bleu toolkit apply smoothing technique chen cherry 2014 introduce smoothing technique improve standard bleu tech- nique geometric evaluation approach label smoothing efficient method improve performance transformer model training phase negative impact perplexity force model uncertain turn positive effect accuracy example suppose predict mask word following sequence cat mask milk imagine output come softmax vector candidate_words=[drink like enjoy appreciate candidate_softmax=[0.7 0.1 0.1,0.1 brutal approach label smoothing system open minded introduce epsilon = ε number element candidate_softmax k=4 label smoothing set ε 0.25 example approach label smoothing straightforward function reduce value candidate_one_hot 1 ε increase 0 value 0 + obtain following result apply approach candidate_smoothed=[0.75,0.083,0.083,0.083 make output open future transfor- mation change transformer use variant label smoothing variant bleu chencherry smoothing chen cherry 2014 introduce interesting way smooth candidate evaluation add ε 0 value chencherry boxing chen + colin cherry meth- let evaluate french english example smoothing reference = je','vous','invite vous lever','pour cette minute de silence candidate = levez','vous','svp','pour cette minute de score = sentence_bleu(reference candidate print("without soothe score score human accept candidate output score weak smooth score 0.37188004246466494 let add openminded smoothing evaluation chencherry = smoothingfunction

r1 = list('je vous invite vous lever pour cette minute de silence candidate = list('levez vous svp pour cette minute de silence sentence_bleu([reference1 reference2 reference3 print("with smooth score",sentence_bleu([r1 candidate smooth machine translation transformer score

reach human acceptability smooth score 0.6194291765462159

see dataset preprocesse bleu evaluate machine translation translation google translate google translate https://translate.google.com/ provide ready use interface trans- lation google progressively introduce transformer encoder translation algorithm follow section implement transformer model translation task ai specialist require enter sentence analyze previous section google translate levez vous svp pour cette minute de silence obtain english translation real time figure 6.2 google translate translation correct industry 4.0 require ai specialist translation task simply web interface de- google provide service require translation google translate platform translation api web developer create interface customer medium translation api translate stream content automl translation service train custom model specific domain google translate project require web developer interface subject matter expert sme linguist ai specialist prerequisite industry 4.0 go ai service bother study ai development trans- former important reason industry 4.0 ai specialist real life ai project run unexpected problem example google translate fit specific need matter goodwill project case google trax come handy use google trax ai need ai developer know fourth industrial revolution connect ai project run smoothly require ai expertise solve complex problem example chapter 14 interpret black box transformer models ai devel- opment require implement google translate ready implement translation trax translation trax google brain develop tensor2tensor t2 t deep learning development easier t2 t extension tensorflow contain library deep learning model contain t2 t good start google brain produce trax end end deep learning library trax contain transformer model apply translation google brain team presently maintain trax section focus minimum function initialize english german problem describe vaswani et al 2017 illustrate transformer performance preprocessed english german dataset transformer ar- chitecture language agnostic begin instal module need google brain trax easy install run import basic trax instal line @title instal trax machine translation transformer import numpy np pip install -q -u trax yes simple let

numpy np pip install -q -u trax yes simple let create transformer model create original transformer model create original transformer model describe chapter 2 getting start architecture transformer model trax function retrieve pretrained model configuration line code @title create transformer model pre trained model config gs://trax ml models translation ende_wmt32k model = trax.models transformer n_heads=8 n_encoder_layers=6 n_decoder_layers=6 model transformer encoder decoder stack stack contain 6 layer 8 head d_model=512 architecture original transformer transformer require pretrained weight run initialize model pretrained weight pretrained weight contain intelligence transformer weight constitute transformer representation language weight express number param- eter produce form machine intelligence iq let life model initialize weight @title initialize model pre trained weight machine configuration intelligence ready run let tokenize sentence tokenize sentence machine translator ready tokenize sentence notebook use vocabulary pre- process trax preprocessing method similar describe chapter preprocesse wmt dataset section sentence tokenize @title tokenize sentence sentence = machine machine intelligence

tokenize = list(trax.data.tokenize(iter([sentence operate program decode sentence produce translation decode transformer transformer encode sentence english decode german model weight constitute set ability trax decoding function intuitive use @title decoding transformer tokenize = tokenized[none add batch dimension tokenized_translation = trax.supervised.decoding.autoregressive_sample model tokenize temperature=0.0 high temperature diverse note high temperature produce diverse result human translator explain chapter define machine translation section finally program de tokenize display translation de tokenize display translation google brain produce mainstream disruptive intuitive implementation trans- trax machine translation transformer program de tokenize display translation line @title de tokenize display translation tokenized_translation = tokenized_translation[0][:-1

remove batch translation = trax.data.detokenize(tokenized_translation output impressive sentence machine machine intelligence translation ich bin nur eine maschine aber ich habe transformer translate machine intelligence maschinenübersicht deconstruct maschinenübersicht maschin machine + übersicht intelligence über literally mean sicht mean sight view transformer tell machine vision machine intelligence grow- e transformers human intelligence machine learn language intelligence conclude experiment google trax chapter go additional essential aspect original transformer start define machine translation human translation set extremely high baseline machine reach see english french english german translation imply numerous problem solve transformer tackle problem set state art bleu record beat preprocesse wmt french english dataset european parliament require cleaning transform dataset line clean datum reduce dataset size suppress word occur frequency threshold machine translation nlp model require identical evaluation method train model wmt dataset require bleu evaluation see geometric assessment good basis score translation modify bleu limit add smooth technique enhance bleu see google translate provide standard translation api medium streaming api custom automl model training service implement google translate api require ai development project roll smoothly hand dirty like old day implement english german translation transformer trax google brain end- end deep learning library cover main building block construct transformer architecture pretraining training preprocessing dataset evaluation method chapter rise suprahuman transformers gpt-3 engines discover mind blow way implement transformer building block explore machine translation exceed human baseline true false machine translation require large dataset true false need compare transformer model dataset true false bleu french word blue acronym nlp metric true false smoothing technique enhance bert true false german english english german machine translation true false original transformer multi head attention sub layer 2 head true false 8 original transformer encoder 6 layer true false original transformer encoder 6 layer 2 decoder layer true false 10 train transformer decoder true false

machine translation transformer english german bleu score reference paper code https://paperswithcode 2014 workshop machine translation wmt https://www.statmt.org/wmt14/ european parliament proceedings parallel corpus 1996 2011 parallel corpus french english jason brownlee ph.d. prepare french english dataset machine translation kishore papineni salim roukos todd ward wei jing zhu 2002 bleu method automatic evaluation machine translation https://aclanthology.org/p02-1040.pdf jason brownlee ph.d. gentle introduction calculate bleu score text python boxing chen colin cherry 2014 systematic comparison smoothing technique sentence level bleu http://acl2014.org/acl2014/w14-33/pdf/w14-3346.pdf ashish vaswani noam shazeer niki parmar jakob uszkoreit llion jones aidan n. gomez lukasz kaiser illia polosukhin 2017 attention need https://arxiv.org/ trax repository https://github.com/google/trax trax tutorial https://trax-ml.readthedocs.io/en/latest/ join book discord space join book discord workspace monthly ask session author rise

suprahuman transformers gpt-3 2020 brown et al 2020 describe training openai gpt-3 model contain 175 billion parameter learn huge dataset 400 billion byte pair encode token extract common crawl datum openai run training microsoft azure super- computer 285,00 cpu 10,000 gpu machine intelligence openai gpt-3 engine supercomputer lead brown et al 2020 zero shot experiment idea use train model downstream task train parameter goal train model directly multi task production api perform task train era suprahuman cloud ai engine bear openai api require high level software skill ai knowledge wonder term suprahuman discover gpt-3 engine perform task human case moment essential understand gpt model build run appreciate magic chapter examine architecture evolution size transformer model investigate zero shot challenge train transformer model little fine tuning model parameter downstream task explore innovative architecture gpt transformer model openai provide specially train version model name engine rise suprahuman transformer gpt-3 engine use 345 m parameter gpt-2

transformer tensorflow openai repository hand dirty understand gpt model interact model produce text completion general conditioning sentence continue 117 m parameter customize gpt-2 model tokenize high level conceptual kant dataset train roberta model chapter 4 pretraine roberta model scratch chapter explore gpt-3 engine require data scientist ar- tificial specialist experienced developer start mean data scientist ai specialist require line gpt-3 engine require fine tuning run google colab notebook fine tune gpt-3 ada engine chapter end new mindset skillset industry 4.0

ai specialist end chapter know gpt model build use seamless gpt-3 api understand gratifying task industry 4.0 ai specialist accomplish 2020 chapter cover following topic getting start gpt-3 model architecture openai gpt model define zero shot transformer model path shot shot build near human gpt-2 text completion model implement 345 m parameter model run interact gpt-2 standard model train language modeling gpt-2 117 m parameter model import customized specific dataset encode customize dataset condition model condition gpt-2 model specific text completion task fine tune gpt-3 model role industry 4.0 ai specialist let begin journey explore gpt-3 transformer model suprahuman nlp gpt-3 transformer model gpt-3 build gpt-2 architecture fully train gpt-3 transformer foun- dation model foundation model task train gpt-3 completion apply nlp task programming task long company realize need data scientist ai specialist start nlp project api like openai provide bother tool openai api available access efficient transformer model train powerful supercomputer world develop tool download library use tool api exist deep pocket good research team world design google openai answer question simple easy start gpt-3 engine start formula 1 indy 500 race car problem try drive car nearly impossible month training gpt-3 engine powerful ai race car run click master incredible horsepower require knowledge acquire beginning book discover need understand architecture gpt model developer ai specialist data scientist fit era suprahuman nlp model architecture openai gpt transformer model transformers go training fine tuning finally zero shot model year end 2017 2020 zero shot gpt-3 transformer model require fine

tuning train model parameter update downstream multi task open new era nlp nlu task gpt-3 fully train transformer model qualify founda- tion model gpt-3 doubt lead powerful openai model google produce foundation model google bert version train supercomputer foundation model represent new way think ai rise suprahuman transformer gpt-3 engine section

new way think ai rise suprahuman transformer gpt-3 engine section learn motivation openai team design gpt model begin go fine tuning zero shot model condition transformer model generate mind blow text completion finally explore architecture gpt model creation process openai team rise billion parameter transformer model speed transformer go small model train nlp task model require little fine tuning staggering

vaswani et al 2017 introduce transformer surpass cnns rnns bleu task radford et al 2018 introduce generative pre training gpt model perform downstream task fine tuning devlin et al 2019 perfect fine tuning bert model radford et al 2019 go gpt-2 model brown et al 2020 define gpt-3 zero shot approach transformer require time wang et al 2019 create glue benchmark nlp model transformer model evolve quickly surpass human baseline wang et al 2019 2020 rapidly create superglue set human baseline high nlu nlp task challenging transformer rapidly progress surpass human baselines superglue leaderboard time writing happen quickly look aspect model size understand evolution happen increase size transformer model 2017 2020 number parameter increase 65 m parameter original transformer model 175b parameter gpt-3 model show table 7.1 vaswani et al 2017 vaswani et al 2017 devlin et al 2019 devlin et al 2019 radford et al 2019 radford et al 2019 radford et al 2019 brown et al 2020 table 7.1 evolution number transformer parameter table 7.1 contain main model design short time date publica- tion come date model actually design author update paper example original transformer set market motion transformer emerge google brain research openai facebook ai produce new model parallel furthermore gpt-2 model large small gpt-3 model example gpt- 3

small model contain 125 m parameter small 345 m parameter gpt-2 model size architecture evolve time number layer model go 6 layer original transformer 96 layer gpt-3 model number head layer go 8 original transformer model 96 gpt-3 model context size go 512 token original transformer model 12,288 architecture size explain gpt-3 175b 96 layer produce impressive result gpt-2 1,542 m 40 layer parameter model comparable number layer double let focus context size understand aspect rapid evolution transformer context size maximum path length cornerstone transformer model reside attention sub layer turn key prop- erty attention sub layer method process context size context size main way human machine learn language large context size understand sequence present drawback context size distance take understand word refer path take analyze long term dependency require change recurrent at- rise suprahuman transformer gpt-3 engine follow sentence require long path find pronoun refer house small fit big couch large table furniture like tiny space think stay time finally decide sell meaning explain long path word house beginning sentence path machine order function define maximum path length sum show table 7.2 big o notation maximum path length table 7.2 maximum path length vaswani et al 2017 optimize design context analysis original transformer model attention bring operation token operation fact layer identical make easy scale size transformer model gpt-3 model size 100 context window maximum length path size 10 context window example recurrent layer rnn store total length context step step maximum path length context size maximum length size rnn process context size gpt-3 model 0(n time long furthermore rnn can- split context 96 head run parallelized machine architecture distribute operation 96 gpu example flexible optimize architecture transformer lead impact vaswani et al 2017 train state art transformer model 36 m sentence brown et al 2020 train gpt-3 model

36 m sentence brown et al 2020 train gpt-3 model 400 billion byte pair encode token extract common crawl datum train large transformer model require machine power available small number team world take total 2.14 10 23

flop brown et al 2020 train gpt-3 175b. design architecture transformer require highly qualify team fund small number organization world size architecture continue evolve probably increase trillion parameter model near future supercomputer continue provide necessary resource zero shot model achieve fine tuning zero shot model start openai research team lead radford et al 2018 want transformer train model gpt model goal train transformer unlabeled datum let attention layer learn language unsupervised datum smart instead teach transformer specific nlp task openai decide train transformer learn language openai want create task agnostic model begin train transformer model raw datum instead rely label datum specialist label datum time consume considerably slow transformer training process step start unsupervised training transformer model fine tune model supervise learning openai opt decoder transformer describe stacking decoder layer section metric result convincing quickly reach level good nlp model fellow nlp research lab promising result version gpt transformer model soon lead radford et al 2019 come zero shot transfer model core philosophy continue train- e gpt model learn raw text take research step focus language modeling example unsupervised distribution examples=(x1 x2 x3 xn example compose sequence symbol sequences=(s1 s2 s3 sn lead metamodel express probability distribution type input goal generalize concept type downstream task train gpt model understand language intensive training rise suprahuman transformer gpt-3 engine gpt model rapidly evolve 117 m parameter 345 m parameter size 1,542 m parameter 1,000,000,000 + parameter transformer bear fine tuning sharply reduce result reach state art metric encourage openai brown et al 2020 go assump- tion conditional probability transformer model train depth able produce excellent result little fine tuning downstream task

openai reach goal train model run downstream task directly fine tuning phenomenal progress describe phase fine tuning ft mean perform sense explore pre- vious chapter transformer model train fine tune downstream task radford et al 2018 design fine tuning task openai team

reduce number task progressively 0 follow step shot fs represent huge step forward gpt train model need inference present demonstration task perform condi- tioning conditioning replace weight updating gpt team exclude process apply conditioning model context provide obtain text completion notebook chapter shot 1s take process train gpt model present demonstration downstream task perform weight updating permit zero shot zs ultimate goal train gpt model present demon- stration downstream task perform approach level efficiency openai gpt team work hard produce state art transformer model explain motivation lead architecture gpt model teach transformer model learn language extensive training focus language modeling context conditioning transformer take context generate text completion novel way instead consume resource learn downstream task work understand input make inference matter task find efficient way train model mask portion input sequence force transformer think machine intelligence machine intelligence human efficient understand motivation lead architecture gpt model let look decoder layer gpt model stack decoder layer understand openai team focus language modeling make sense mask attention sublayer choice retain decoder stack exclude encoder stack brown et al 2020 dramatically increase size decoder transformer model excellent result gpt model structure decoder stack original transformer design vaswani et al 2017 describe decoder stack chapter 2 getting start architecture transformer model necessary minute ar- chitecture original transformer gpt model decoder architecture show figure 7.1 figure 7.1 gpt decoder architecture rise suprahuman transformers gpt-3

engine recognize text position embedding sub layer mask multi head self atten- tion layer normalization sub layer feedforward sub layer output addition version gpt-2 text prediction task classification openai team customize tweak decoder model model radford et al 2019 present few gpt model brown et al 2020 describe few gpt-3 175b model reach unique size require computer resource team world access nparam = 175.0b nlayer = 96 dmodel = 12288 nhead = 96 let look grow number gpt-3 engine gpt-3 model train accomplish specific task

different size list engine available time document openai https://beta.openai.com/docs/engines base series engine different function example davinci engine analyze complex intent curie engine fast good summarization babbage engine good semantic search ada engine good parse text openai produce engine market instruct series provide instruction base description example available gpt-3 example section chapter codex series translate language code explore series chapter 16 emergence transformer driven copilots content filter series filter unsafe sensitive text explore series chapter 16 emergence transformer driven copilot explore process lead fine tuning zero shot gpt-3 model see gpt-3 produce wide range engine time source code gpt model build gpt-3 transformer model source code publicly available time gpt-2 model sufficiently powerful understand inner working gpt model ready interact gpt-2 model train use train gpt-2 345 m model text completion 24 decoder layer self attention sublayer 16 head train gpt-2 117 m model customize text completion 12 decoder layer self attention layer 12 head let start interact pretrained 345 m parameter gpt-2 model generic text completion gpt-2 explore example gpt-2 generic model goal example run determine level abstract reasoning gpt model attain section describe interaction gpt-2 model text completion focus step 9 openai_gpt_2.ipynb notebook describe detail appendix iii generic text completion gpt-2 let understand specific example

pretrained gpt-2 apply step 9 interact gpt-2 section interact gpt-2 345 m model interact model run interact_model cell @title step 9 interact gpt-2 read section generic pretrained gpt-2 model react specific example read appendix iii generic text completion gpt- 2 detail generic gpt-2 model implement google read appendix iii directly contain interaction step 9 de- rise suprahuman transformers gpt-3 engine prompt enter context figure 7.2 context input text completion try type context wish standard gpt-2 model try sentence write immanuel kant human reason sphere cognition call consider question decline present nature answer transcend faculty mind press enter generate text output relatively random gpt-2 model train dataset run stochastic model let look line gpt model generate time run grant conception

peculiarity second law logic experience measure end apprehension close consciousness solution scholastic perplexity moral religious impossible existence blasphemous stop cell double click run button cell press ctrl + m stop generate text transform code text copy program cell output rich observe fact context enter condition output generate model context demonstration model learn context modify parameter text completion condition context open door transformer model require fine tuning semantic perspective output interesting grammatical perspective output convincing well follow section present interaction custom text completion train custom gpt-2 language model continue approach section explore example gpt-2 custom model train specific dataset goal remain determine level abstract reasoning gpt model attain section describe interaction gpt-2 model text completion train specific dataset focus step 12 training_openai_gpt_2.ipynb notebook describe detail appendix iv custom text completion gpt-2 let understand interaction gpt-2 improve train step 12 interactive context completion example run conditional sample context enter condition model think want complete text generate tailor paragraph run cell explore magic @title step 12 interactive context completion example import os import runtime restart python interactive_conditional_samples.py --temperature 0.8 --top_k 40 necessary minute step

restart python interactive_conditional_samples.py --temperature 0.8 --top_k 40 necessary minute step 9 interact gpt-2 appendix iii generic text completion gpt-2 difference response program prompt enter context figure 7.3 context input text completion read section example custom gpt-2 model improve response read appendix iv custom text completion gpt-2 understand train gpt-2 obtain specific response decide read appendix iv directly contain interaction step 12 describe rise suprahuman transformer gpt-3 engine let enter paragraph write immanuel kant step 9 generic text completion gpt-2 section chapter human reason sphere cognition call consider question decline present nature answer transcend faculty mind press enter generate text previously structured logical output change run make transformer attractive time result random impressive let look line gpt-2 model produce nature reason kind

power mind call think understand reason thing apprehend seek understand object mind apprehend apprehend thing speak reason understanding understanding human reason knowledge understanding natural faculty understand human reason natural faculty reason understanding human reason see know thing natural sense natural sense understand source knowledge understand reason thing understand understand thing comprehend understand mean reason thing understand know thing understand stop cell double click run button cell press ctrl + m.

wow doubt anybody difference text completion produce train gpt-2 model human generate different output run fact think model outperform human abstract exercise philosophy reason logic draw conclusion experiment train transformer model produce text completion human level gpt-2 model reach human level text generation complex abstract text context efficient way condition model demonstrate expect text completion text generation base text conditioning context sentence enter conditioning text context example experiment text completion train model datum replace content dset.txt file datum happen remember train gpt-2 model react like human enter short incomplete uninteresting tricky context obtain puzzled bad result gpt-2 expect good real life let gpt-3 playground train gpt-3 react example test run openai gpt-3 task section run gpt-3 different way run gpt-3 task online code implement gpt-3 google colab notebook let begin run nlp task online run nlp task online industry 4.0 example api directly ask gpt-3 let define standard structure prompt response n = nlp task input e = explanation gpt-3 engine e precede t input t = text content wish gpt-3 look input s = show gpt-3 expect s follow t add necessary input r = gpt-3 response output gpt-3 engine book sign gpt-3 api openai give free budget start free budget cover cost cost run example book twice rise suprahuman transformer gpt-3 engine structure prompt describe guideline gpt-3 flexible variation possible ready run educational example online api question answer q&a exist knowledge e = q t = president united states 1965 s = r = prompt response q president united states 1965 lyndon b. johnson president united states 1965 q

human moon neil armstrong human moon movie emoji e = example movie title t = s = implicit example r = example emoji prompt response new prompt response summarize second grader summarize 2nd grader e = second grader ask passage mean t = initial conclusion s = rephrase plain language second grader understand r = summary prompt response second grader ask passage mean initial conclusion divide category fact fiction fact openai powerful nlp service world main fact openai engine powerful zero shot require hunting kind transformer model pre training fine tuning supercomputer train model unique prompt design obtain surprisingly accurate response implement nlp task section require copy paste action software beginner perform fiction begin dystopian hype assertion ai replace datum scientist ai specialist true answer question ask following question example run know sentence incorrect place know answer correct human read confirm engine know grammar correction task response incorrect understand happen help improve prompt revert manual mode well- design human interface truth human need intervene answer question manually rule basis quality control automate pipeline tool fact convincing true run nlp task require little development fiction convincing human require openai engine replace human help perform high level gratify task fly jet have build rephrase plain language second grader understand conclusion openai powerful nlp service world important fact openai engine powerful zero shot require hunting kind transformer model pre training fine tuning supercomputer train model unique prompt design surprisingly result impressive huge step forward history nlp rise suprahuman transformer gpt-3 engine number nlp task implement zero shot gpt-3 engine provide openai endless control expect transformer model true industry 4.0 ai guru hand dirty implement ready use api explore architecture openai gpt model build gpt-2 model engine work know gpt model well industry 4.0

nlp expert implement real life project let continue approach drill architecture openai gpt getting start gpt-3 engine openai powerful transformer engine world gpt-3 model perform

hundred task gpt-3 task train section use api getting_started_gpt_3.ipynb use gpt-3 openai website https://openai.com/ sign openai playground everybody try like google translate user friendly online service let try task run nlp task gpt-3 let start gpt-3 step google colab open getting_started_gpt_3.ipynb chapter directory book github need change setting notebook api need local computing power task section step section one notebook run nlp simple step step 1 instal openai install openai following command pip install openai openai instal restart runtime message indicate show follow output restart runtime run cell sure openai import step 2 enter api key api key give python c java option python section openai.api_key=[your api key update cell api key os.environ['openai_api_key = your_key key variable openai.api_key = os.getenv("openai_api_key let run nlp task step 3 run nlp task default parameter copy paste openai example grammar correction task response = openai.completion.create prompt="original go market.\nstandard american rise suprahuman transformer gpt-3 engine task correct grammar mistake go market process response wish parse openai response dictionary object openai object contain detailed information task ask object display display response object explore object text market create number d model vary run ask object dictionary display text print process output display response object r = response["choices"][0 output text dictionary grammatically correct sentence market nlp task example cover industrial approach gpt-3 engine usage example openai provide interactive educational interface require api school teacher consultant linguist philosopher anybody wish use gpt-3 engine educational purpose experience ai begin api notebook getting_started_gpt_3.ipynb begin explore getting start gpt-3 engine section chapter experiment grammar correction open notebook step 4 example 1 grammar correction step 6 run nlp task custom parameter response = openai.completion.create defult engine

davinci default prompt task:"original prompt="original go market.\n standard american request body limit prompt body contain key parameter engine="davinci choice openai gpt-3 engine use possibly model future temperature=0 high value 0.9 force model risk modify

temperature top_p time max_tokens=60 maximum number token response rise suprahuman transformers gpt-3 engine top_p=1.0 way control sampling like temperature case top_p percentage token probability mass consider 0.2 system 20 probability mass frequency_penalty=0.0 value 0 1 limit frequency token presence_penalty=0.0 value 0 1 force system use new token produce new idea stop=["\n signal model stop produce new token parameter describe source code level step 7b-8 import define model section appendix iii generic text completion gpt-2 play parameter gpt-3 model gain access gpt-2 model appendix iii generic text completion gpt-2 concept case section focus prompt prompt="original go market.\n standard american english prompt divide part original signal model follow original text model go market.\n prompt show model original text standard american english show model task expect let far change task standard american english produce prompt="original go market.\n standard american english text response text market fine want contraction sentence english contraction produce prompt="original go market.\n english contraction text response text market wow impressive let try language french contraction produce text elle n'est pas all\u00e9e au march\u00e9 impressive \u00e9 simply need post process é option possible industry 4.0 cross disciplinary imagination limit gpt-3 example openai contain example openai provide online playground explore task openai provide source code example https://beta.openai.com/examples click example grammar example explore grammar correction figure 7.4 grammar correction section openai openai describe prompt sample response task figure 7.5 sample response correct prompt rise suprahuman transformers gpt-3 engines choose playground run online chapter run nlp task online section click open playground button figure 7.6 open playground button notebook chapter figure 7.7 run code davinci engine getting_started_gpt_3.ipynb

contain example run practice implement openai gpt-3 example read link explanation provide openai link documen- tation provide cell run cell observe gpt-3 behavior run example notebook example 1 grammar correction example 2 english french translation example 3

instruct series provide instruction example 4 movie emoji example 5 program language language example python javascript warning need obtain special permission openai run example use davinci codex engine code generator example run notebook contact openai request access codex example 6 advanced tweet classifier example 7 q&a example 8 summarize text example 9 parse unstructured datum example 10 calculate time complexity run task example page https://beta.openai.com/examples let compare output gpt-2 gpt-3 compare output gpt-2 gpt-3 curiosity satisfy powerful gpt-3 model produce example submit pretrained gpt-2 model custom train gpt-2 example gpt-2 model human reason sphere cognition call consider question decline present nature answer transcend faculty mind rise suprahuman transformer gpt-3 engine mean thing experience reason reason bad guide insufficient thing reach example existence god result mind blow explain text mean include deep philosophical prove point chapter transformer model attain abstract reasoning help micro decision fast move world openai gpt-3 fully train model gpt-3 fine tune let section show fine tune gpt-3 learn logic transformer need learn logic in- ference entailment understand language human level section gpt-3 train work immanuel kant kantgpt.csv similar file

train bert type model chapter 4 pretraine roberta model scratch master fine tune gpt-3 use type datum teach specific domain knowledge graph text openai provide efficient document service fine tune gpt-3 engine train gpt-3 model different type engine see rise billion parameter transformer model section chapter davinci engine powerful expensive use ada engine expensive produce sufficient result explore gpt-3 experiment fine tune gpt-3 involve phase prepare datum fine tune gpt-3 model fine tuning key make gpt-3 application customize fit need project ticket ai freedom rid application bias teach thing want know leave footprint ai prepare datum open fine_tuning_gpt_3.ipynb google colab github chapter directory openai document datum preparation process detail step 1 instal openai step 1 install import openai pip install openai restart runtime installation complete run cell sure import openai execute install wand visualize log pip install wandb

enter api key step 2 enter api key step 2 enter key step 3 activate openai data preparation module load file section load kantgpt.csv kantgpt.csv raw unstructured file openai inbuilt data cleaner ask question step rise suprahuman transformer gpt-3 engine openai detect file csv file convert jsonl file jsonl contain line plain structured text openai track change approve base analysis perform following action necessary format csv convert jsonl necessary remove 27750 row completion recommended remove 903 duplicate row y n y recommended add suffix separator > prompt y n y recommended remove prefix completion completion y n y recommended add suffix ending \n completion y n y recommended add whitespace character beginning completion y n y openai save convert file kantgpt_prepared.jsonl ready fine tune gpt-3 split notebook separate notebook data preparation step 4 create os environment step 4 fine tuning process create os environment api key step 5 fine tune openai ada engine step 5 trigger fine tune openai ada engine jsonl file save datum openai api fine_tunes.create -t kantgpt_prepared.jsonl -m ada

openai request steam interrupt openai indicate instruction continue fine tune execute openai api fine_tunes.follow -i your_fine_tune step 6 interact fine tune model step 6 interact fine tune model prompt sequence close immanuel kant openai api completions.create -m ada:[your_model info concept priori instruction run completion task your_model info display openai end fine tune task copy paste cell(add run command line insert your_model info follow cell completion convincing concept priori term freedom concept fine tune gpt-3 show importance understand transformer design ai pipeline api let change role ai specialist role industry 4.0 ai specialist nutshell role industry 4.0 developer cross disciplinary ai guru developer data scientist ai specialist progressively learn linguistics business goal subject matter expertise industry 4.0 ai specialist guide team practical cross disciplinary knowledge experience human expert mandatory domain implement transformer moral ethic industry 4.0 ai guru ensure moral ethical practice enforce imple- mente humanlike transformer european

regulation example strict re- quire automate decision explain user necessary anti discrimination law protect citizen automate bias rise suprahuman transformer gpt-3 engine prompt response user ui developer need industry 4.0 ai guru explain create right prompt nlp task transformer model task verify response quality control understand model happen model behave expect tweak hyperparam- eter deeply issue chapter 14 interpret black box transformer model initial conclusion divide category fact fiction fact openai powerful nlp service world fact include openai engine powerful zero shot engine require hunting kind transformer model pre training fine tuning supercomputer train model unique prompt design surprisingly accurate response implement nlp task chapter require copy paste action software beginner perform people believe ai replace datum scientist ai specialist true answer question ask follow question example run chapter know sentence incorrect know answer correct human read confirm engine know grammar correction task response incorrect understand happen help improve prompt revert manual mode

response incorrect understand happen help improve prompt revert manual mode design human interface truth human need intervene answer question manually rule basis quality control automated pipeline tool fact convincing run nlp task transformer require little development case human require openai engine replace human help perform high level gratify task fly jet have build need answer exciting question

bring section let explore new fascinating industry 4.0 role wonderful path future ai let sum chapter exploration chapter discover new era transformer model train billion parameter supercomputer openai gpt model take nlu reach nlp devel- see gpt-3 zero shot model perform nlp task api direct- ly online api online version google translate pave way mainstream online usage ai explore design gpt model build original transformer decoder stack masked attention sub layer continue philosophy left right training how- sheer power calculation subsequent self attention sub layer make implement 345 m parameter gpt-2 model tensorflow goal inter- act

train model far see context provide condition output reach result expect enter specific input kant dataset train 117 m parameter gpt-2 model customize dataset interaction relatively small train model produce fascinating result run nlp task online openai api fine tune gpt-3 model chapter show fully pretrained transformer engine automatically accomplish task little help engineer

mean user need ai nlp developer data scientist ai specialist anymore future instead user simply upload task definition input text cloud transformer model download result mean industry 4.0 data scientist ai specialist evolve pilot powerful ai system increasingly necessary ensure input ethical secure modern age ai pilot understand transformer build adjust hyperparameter ai ecosystem rise suprahuman transformer gpt-3 engine chapter apply transformer legal financial document ai text summariza- tion transformer model limit multi task model explore new frontier zero shot method train parameter true false gradient update perform run zero shot model true false gpt model decoder stack true false impossible train 117 m gpt model local machine true false impossible train gpt-2 model specific dataset true false 6 gpt-2 model condition generate text true false gpt-2 model analyze context input produce completion content 8 interact 345m parameter gpt model machine 8 supercomputer 285,000 cpu exist true false 10 supercomputer thousand gpu game changer ai true false openai gpt-3 engine https://beta.openai.com/docs/engines/engines bertviz github repository jesse vig https://github.com/jessevig/bertviz openai supercomputer https://blogs.microsoft.com/ai/openai-azure- ashish vaswani noam shazeer niki parmar jakob uszkoreit llion jones aidan n. go- mez lukasz kaiser illia polosukhin 2017 attention need https://arxiv.org/ alec radford karthik narasimhan tim salimans ilya sutskever 2018 improve language understanding generative pre training https://cdn.openai.com/research-covers/ jacob devlin ming wei chang kenton lee kristina toutanova 2019 bert pre train- ing deep bidirectional transformers language understanding https://arxiv.org/ alec radford jeffrey wu rewon child david luan dario amodei ilya sutskever 2019 lan- guage models unsupervised

multitask learners https://cdn.openai.com/better- tom b. brown benjamin mann nick ryder melanie subbiah jared kaplany prafulla dhariwal arvind neelakantan pranav shyam girish sastry amanda askell sandhini agarwal ariel herbert voss gretchen krueger tom henighan rewon child aditya ramesh daniel m. ziegler

jeffrey wu clemens winter christopher hesse mark chen eric sigler mateusz litwin scott gray benjamin chess jack clark christopher berner sam mccandlish alec radford ilya sutskever dario amodei 2020 language models shot learners https://arxiv alex wang yada pruksachatkun nikita nangia amanpreet singh julian michael felix hill omer levy samuel r. bowman 2019 superglue stickier benchmark general pur- pose language understanding systems https://w4ngatang.github.io/static/papers/ alex wang yada pruksachatkun nikita nangia amanpreet singh julian michael felix hill omer levy samuel r. bowman 2019 glue multi task benchmark analysis platform natural language understanding https://arxiv.org/pdf/1804.07461.pdf openai gpt-2 github repository https://github.com/openai/gpt-2 n. shepperd github repository https://github.com/nshepperd/gpt-2 common crawl datum https://commoncrawl.org/big-picture/

 transformer legal financial document ai explore architecture training fine tuning usage transformer ecosystem seven chapter chapter 7 rise suprahuman transformer gpt-3 engines discover openai begin experiment zero shot model require fine tuning development implement line underlie concept evolution rely transformer strive teach machine understand language express human like manner go train model teach language machine raffel et al 2019 design transformer meta model base simple assertion nlp problem represent text text function type nlp task require kind text context generate form text response text text representation nlp task provide unique framework analyze transform- er methodology

practice idea transformer learn language transfer learning training fine tuning phase text text approach raffel et al 2019 name approach text text transfer transformer 5 t t5 new model bear begin chapter go concept architecture t5 transformer model apply t5 summarize document hugging face model apply transformer legal financial document ai text summarization finally transpose text text approach context process gpt-3 engine usage mind blow perfect zero shot response exceed human imagine chapter cover following topic text text transformer model architecture t5 model evolution transformer model training learn hug face transformer model implement t5 model summarize legal text summarize financial text limit transformer model step explore text text methodology define raffel et al 2019 design universal text text model google nlp technical revolution start vaswani et al 2017 original transformer 2017 attention need topple 30 + year artificial intelligence belief rnns cnns apply nlp task take stone age nlp nlu 21 st century chapter 7 rise suprahuman transformers gpt-3 engines sum second revolution boil erupt google vaswani et al 2017 original transformer openai brown et al 2020 gpt-3 transformer original transformer focus performance prove attention need nlp nlu task openai second

focus performance prove attention need nlp nlu task openai second revolution gpt-3 focus take transformer model fine- tune pretraine model shot train model require fine tuning second revolution machine learn language apply downstream task human essential perceive revolution understand t5 model represent revolution attention technique second revolution teach machine understand language nlu let solve nlp problem 2019

google think line openai transformer perceive technical consideration abstract level natural language revolution disruptive time settle forget source code machine resource analyze transformer high level raffel et al 2019 design conceptual text text model implement let representation second transformer revolution abstract model rise text text transformer model raffel et al 2019 set journey pioneer goal explore limit transfer learning unified text text

transformer google team work approach empha- size modify original transformer fundamental architecture start point raffel et al 2019 want focus concept technique show interest produce late transformer model call silver bullet transformer model n parameter layer time t5 team want find good transformer understand language human learn language apply knowledge wide range nlp task transfer learning core concept t5 model find abstract model thing communicate start sequence follow sequence b b turn start sequence lead sequence show figure 8.1 figure 8.1 sequence sequence representation communication communicate music organize sound communicate danc- ing organized body movement express painting coordinate shape color apply transformer legal financial document ai text summarization communicate language word group word text try understand text pay attention word sentence direction try measure importance term understand sentence focus word query rest keyword sentence determine value attention pay define attention layer transformer second

let sink deceptively simple right take 35 + year topple old belief surround rnns cnns thought process accompany fascinating watch t5 learn progress help think well technical revolution attention layer simultaneously attend token sequence lead t5 conceptual revolution t5 model sum text text transfer transformer nlp task express text text problem solve prefix instead task specific format raffel et al 2019 problem solve unify task specific format idea find way input format task submit transformer way model parameter train type task text text format google t5 team come simple solution add prefix input sequence need thousand additional vocabulary language invention prefix long forget genius example need find word describe prepayment prehistoric precambrian thousand word use pre raffel et al 2019 propose add prefix input sequence t5 prefix tag indicator like cls classification transformer model instead t5 prefix contain essence task transformer need solve prefix convey meaning follow example translate english german + sequence translation chapter 6 machine translation transformer cola sentence + sequence corpus linguistic acceptability cola chapter 3 fine tune bert models fine tune bert transformer model stsb sentence 1:+[sequence semantic textual similarity

benchmark natural language inference entailment similar problem describe chapter 5 down- stream nlp task transformer summarize + sequence text summarization problem solve text summarization t5 section chapter obtain unified format wide range nlp task express figure 8.2 figure 8.2 unify input format transformer model unified input format lead transformer model produce result sequence mat- ter problem solve t5 input output nlp task unify show figure 8.3 figure 8.3 t5 text text framework unification process make possible use model hyperparameter optimizer wide range task go standard text text input output format let look ar- chitecture t5 transformer model apply transformer legal financial document ai text summarization t5 model raffel et al 2019 focus design standard input format

model raffel et al 2019 focus design standard input format obtain text output goo- gle t5 team want try new architecture derive original transformer bert like encoder layer gpt like decoder layer instead team focus define nlp task standard format choose use original transformer model define chapter 2 get start architecture transformer model figure 8.4 figure 8.4 original transformer model t5 raffel et al 2019 keep original transformer architecture term emphasize key aspect slight vocabulary functional change follow list contain main aspect t5 model encoder decoder remain model encoder decoder layer block sublayer subcomponent contain self attention layer feedforward network use word block subcomponent ® -like language allow assemble block piece component build model transformer component standard building block assemble way understand transformer model understand basic building block go chapter 2 get start architecture self attention order independent mean perform operation set see chapter 2 self attention use dot product matrix recurrence explore relationship word sequence positional encoding add word embedding make dot product original transformer apply sinusoidal cosine signal transformer learn position embedding t5 use relative position embedding instead add- e arbitrary position input t5 positional encoding rely extension self attention comparison pairwise relationship

shaw et al 2018 reference section chapter positional embedding share evaluate layer model define standardization input t5 transformer model text- let use t5 summarize document text summarization t5 nlp summarize task extract succinct part text section start present hugging face resource use chapter initialize t5 large transform- er model finally use t5 summarize document include legal let begin introduce hugging face framework hugging face design framework implement transformers high level hug- ging face fine tune bert model chapter 3 fine tune bert models train roberta model chapter 4 pretraine roberta model scratch apply transformer legal financial document ai text summarization expand knowledge need explore approach trax chapter 6 machine translation transformer openai model chapter 7 rise supra-human transformer gpt-3 engine chapter use hugging face framework explain online resource end chapter unique potential gpt-3 engine hugging face provide primary resource framework model dataset hugging face transformer resource subsection choose t5 model implement chapter wide range model find hugging face model page figure 8.5 figure 8.5 hugging face model page https://huggingface.co/models search model case look t5 large model smoothly run google colaboratory type t5 search t5 model obtain list t5 model choose figure 8.6 search t5 model original t5 transformer available base baseline model design similar bertbase 12 layer 220 million parameter small small model 6 layer 60 million parameter large design similar bertlarge 12 layer 770 million parameter 3b 11b use 24 layer encoder decoder 2.8 billion 11 billion description bertbase bertlarge minute later review model chapter 3 fine tune bert models apply transformer legal financial document ai text summarization case select t5 large figure 8.7 use hugging face model figure 8.7 show use model code write look list file model basic configuration file look configuration file initialize model initialize t5 large transformer model section chapter hugging face provide dataset

t5 large transformer model section chapter hugging face provide dataset metric dataset train test model https://huggingface.co/ metric resource measure performance model https:// dataset metric classical aspect nlp chapter implement dataset metric instead focus implement

text summarize let start initialize t5 transformer model initialize t5 large transformer model subsection initialize t5 large model open follow notebook summarizing text_with_t5.ipynb find directory chapter github let start t5 getting start t5 subsection install hugging face framework initialize t5 model install hugging face transformer pip install transformer pin version 0.1.94 sentencepiece notebook hugging face stable pip install sentencepiece==0.1.94

hugging face github repository clone hugging face framework provide range high level transformer function implement choose display architecture model initialize model set display_architecture true structure encoder layer decoder layer feedforward sublayer display program import torch json work transformer mean open transformer architecture framework research lab share recommend pytorch tensorflow possible environment matter level abstraction transformer model specific task model zero shot model overall performance let import tokenizer generation configuration class transformer import t5tokenizer t5forconditionalgeneration t5config use t5 large model select t5 model hugging face list go chapter hugging face section note hugging face transformer continually evolve update library module adapt market default version work pin pip install transformers==[version run function notebook apply transformer legal financial document ai text summarization import t5 large conditional generation model generate text t5 large model = t5forconditionalgeneration.from_pretrained('t5 large tokenizer = t5tokenizer.from_pretrained('t5 large initialize pretrained tokenizer take line prove toke- nize dictionary contain vocabulary need investigate relation tokenizer dataset chapter 9 matching tokenizer dataset program

initialize torch.device cpu cpu notebook torch.device object device torch tensor allocate device = torch.device('cpu ready explore architecture t5 model explore architecture t5 model subsection explore architecture configuration t5 large model display_architecture==true configuration model example basic parameter model model t5 transformer 16 head 24 layer

text text implementation t5 add prefix input sentence trigger task perform prefix make possible represent wide range task text- text format modify model parameter case prefix summarization prefix summarize t5 implement beam search algorithm expand significant text apply early stopping num_beam sentence complete batch make sure repeat ngram equal no_repeat_ngram_size control length sample min_length max_length apply length penalty interesting parameter vocabulary size vocabulary size topic vocabulary lead sparse representation hand little vocabulary distort nlp task explore chapter 9 matching tokenizer dataset detail transformer stack simply print model example peek inside block layer encoder stack number 0 23 q linear(in_features=1024 out_features=1024 bias = false k linear(in_features=1024 out_features=1024 bias = false v linear(in_features=1024 out_features=1024 bias = false o linear(in_features=1024 out_features=1024 bias = false apply transformer legal financial document ai text summarization dropout dropout(p=0.1 inplace = false wi linear(in_features=1024 out_features=4096 wo linear(in_features=4096 out_features=1024 dropout dropout(p=0.1 inplace = false dropout dropout(p=0.1 inplace = false

model run operation 1,024 feature attention sublayer 4,096 inner calculation feedforward network sublayer produce output 1,024 feature symmetrical structure transformer maintain layer minute encoder stack decoder stack attention sublayer feedforward sublayer choose select specific aspect model run cell wish initialize t5 transformer let summarize document summarize document t5 large section create summarize function text wish sum- marize summarize legal financial example finally define limit start create summarization function create summarization function let create summarize function name summarize way send text want summarize function function take parameter parameter preprocess_text text summarize second parameter ml maximum length summarize text parameter variable send function time hug face provide ready use summarizing function recom- mend learn build function customize critical task necessary context text ground truth strip \n character preprocess_text =

text.strip().replace("\n apply innovative t5 task prefix summarize input text t5_prepared_text = summarize + preprocess_text t5 model unified structure task prefix + input sequence approach simple take nlp transformer model close universal training zero shot downstream task display process strip prepare text task prefix print preprocesse prepare text \n t5_prepared_text simple right take 35 + year rnns cnns transformer take bright research team world transformer design specific task multi task model require little fine tuning finally google research team create standard format transformer input text contain prefix indicate nlp problem solve feat output display contain preprocesse prepare text preprocesse prepare text summarize united states declaration independence summarize prefix indicate task solve apply transformer legal financial document ai text summarization text encode token id return torch

tensor tokenized_text = tokenizer.encode(t5_prepared_text return_tensors="pt encode text ready send model generate summary parameter describe get start t5 section summary_id = model.generate(tokenized_text number beam remain model import no_repeat_ngram size bring 2 instead 3 generate output decode tokenizer output = tokenizer.decode(summary_ids[0 skip_special_tokens = true import initialize define summarization function let experiment t5 model general topic general topic sample subsection run text write project gutenberg t5 model use sample run test summarizing function copy paste text wish load text add code load dataset choice summary loop program goal chapter run sample t5 work input text beginning project gutenberg e book contain declaration independence united states america united states declaration independence etext release project gutenberg early 1971 title store email instruction set require tape diskpack hand mount retrieval diskpack size large cake cake carrier cost $ 1500 contain 5 megabyte file take 1 2 tape backup keep plus paper tape 10,000 file hope online end 2001 1 2 comparably price drive 2001 summarize function send text want summarize maximum length summary print("number characters:",len(text print \n\nsummarized text \n",summary output show send 534 character original text ground truth preprocesse summary prediction number character 534 preprocesse prepare text summarize

united states declaration independence united states declaration independence etext publish project gutenberg early 1971 10,000 file hope online end of2001 1 2

comparably price drive 2001 united states declaration independence etext release project gutenberg early 1971 let use t5 difficult summary bill rights sample follow sample take bill rights difficult express special right person bill rights v person shall hold answer capital infamous presentment indictment grand jury case land naval force militia actual service apply transformer legal financial document ai text summarization time war public danger shall person subject offense twice jeopardy life limb shall compel criminal case witness deprive life liberty property process law shall private property take public use print("number characters:",len(text print \n\nsummarized text \n",summary remember transformer stochastic algorithm output vary time run say t5 summarize input text simply shorten number character 591 preprocessed prepare text summarize person shall hold answer person shall hold answer capital infamous crime case arisingin land naval force militia actual service time war public danger sample significant show limit transformer model nlp model face face text present sample work user believe transformer solve nlp challenge face matter innovative maybe provide long text summarize parameter large model change structure t5 model matter hard try sum- marize complex text nlp model find document model fail model fail task humble admit superglue human baseline difficult beat need patient work hard improve transformer model perform well today room lot progress raffel et al 2018

choose appropriate title describe approach t5 explore limit transfer learning unified text text transformer necessary time experiment example find legal document explore limit transfer learning modern day nlp pioneer discover exciting result find area need improvement let try corporate law sample corporate law sample corporate law contain legal subtlety make summarize task tricky input sample excerpt corporate law state montana usa montana corporate law text = law corporation prescribe corporation incorporate state montana serve

lawful purpose state montana corporation power natural person carry business activity corporation sue sue corporate perpetual succession corporation buy sell acquire interest real personal property conduct business carry operation office exercise power state territory district possession u.s. foreign country appoint officer agent corporation duty fix compensation corporation contain word corporation abbreviation corp corporation deceptively similar corporation incorporate state deceptively identical fictitious adopt foreign corporation have business transaction state corporation form natural person execute file article incorporation secretary state filing qualification director fix article incorporation bylaw name address initial director purpose incorporation set forth article incorporation article incorporation contain corporate number share authorize issue brief statement character business carry corporation name address director successor elect address incorporator shareholder power change size board director apply transformer legal financial document ai text summarization print("number characters:",len(text print \n\nsummarized text \n",summary result satisfying number character 1816 preprocessed prepare text summarize law corporation prescribe corporation incorporate state montana serve lawful purpose corporation sue sue corporate perpetual succession conduct business carry operation office time t5 find essential aspect text summarize time incorporate sample happen play parameter affect implement t5 summarize text time experiment summarization openai gpt-3 engine summarization gpt-3 essential understand architecture t5 transformer gpt-3 engine behave text goal benchmark company model goal industry 4.0 ai guru broad knowledge nlp https://openai.com/

sign sign example page select summarize 2nd grader figure 8.8 gpt-3 example page window open enter prompt submit text t corporate sample previous section gpt-3 model prompt p = e + t + s e tell model explanation simple second grader ask passage mean text t previous section quote law corporation prescribe corporation incorporate state montana serve lawful purpose state montana corporation power natural person carry business activity shareholder power

change size board director s show type task expect rephrase plain language second grader understand response produce gpt-3 davinci engine convincing corporation business corporation person corporation people corporation sue corporation buy sell trade thing corporation different name people corporation form people want start business file paper government official t modify e s e change mba college student ask passage mean s change rephrase mba college language apply transformer legal financial document ai text summarization gpt-3 engine generate response previous request add useful information college student corporation director charge make decision business director elect shareholder shareholder change size board director gpt-3 model convincing represent rise power cloud ai deeply summarize prompt chapter 16 emergence transformer drive copilot explore chapter see t5 transformer model standardize input encoder decoder stack original transformer original transformer architecture identical structure block layer encoder decoder stack original transformer standardized input format nlp task raffel et al 2018 design standard input wide range nlp task define text to- text model add prefix input sequence indicate nlp problem type solve lead standard text text format text text transfer transformer t5 bear see deceivingly simple evolution possible use model hyperparameter wide range nlp task invention t5 take standardization process transformer model step implement t5 model summarize text test model text ready use training dataset test model constitutional corporate sample result interesting discover limit transformer model predict raffel et al 2018

interesting discover limit transformer model predict raffel et al 2018 finally explore tremendous power gpt-3 engine methodology calculation efficiency show transformer brilliant approach have powerful trans- engine world help attain effective perfect result goal benchmark company model industry 4.0 ai guru deep understanding transformer chapter chapter 9 matching tokenizers datasets explore limit tokenizer define method possibly improve nlp task t5 model encoder stack like bert model true false t5 model encoder decoder stack true false t5 model use relative positional encoding absolute positional encoding true

false text text model design summarization true false text text model apply prefix input sequence determine nlp task t5 model require specific hyperparameter task true false advantage text text model use hyperparameter nlp task true false 8 t5 transformer contain feedforward network true false hug face framework make transformer easy implement true false 10 openai transformer engine game changer true false colin raffel noam shazeer adam roberts katherine lee sharan narang michael matena yanqi zhou wei li peter j. liu 2019 explore limit transfer learning

unified text text transformer https://arxiv.org/pdf/1910.10683.pdf ashish vaswani noam shazeer niki parmar jakob uszkoreit llion jones aidan n. go- mez lukasz kaiser illia polosukhin 2017 attention need https://arxiv.org/ peter shaw jakob uszkoreit ashish vaswani 2018 self attention relative position hugging face framework resources https://huggingface.co/ u.s. legal montana corporate laws https://corporations.uslegal.com/state- declaration independence united states america thomas jefferson https:// united states bill rights united states https://www.gutenberg.org/ebooks/2 apply transformer legal financial document ai text summarization join book discord space join book discord workspace monthly ask session author matching tokenizers study transformer model tend focus model architecture dataset provide train explore original transformer fine tune bert like model train roberta model explore gpt-3 model train gpt-2 model implement t5 model go main benchmark task dataset train roberta tokenizer tokenizer encode datum explore limit tokenizer evaluate fit model build ai data drive raffel et al 2019 like author cite book spend time prepare dataset transformer model chapter limit tokenizer hinder quality downstream transformer task pretrained tokenizer face value specific dictionary word use advanced medical language example word process generic pretrained tokenizer start introduce tokenizer agnostic good practice measure quality tokenizer describe basic guideline dataset tokenizer tokenization limit tokenizer word2vec tokenizer describe problem face tokenizing method limit illustrate python program continue investigation run gpt-2 model dataset contain specific vocabulary

unconditional conditional sample match tokenizer dataset limit byte level bpe method build python program display result produce gpt-2 tokenizer problem occur datum encoding process superiority gpt-3 necessary common nlp analysis end chapter probe gpt-3 engine speech pos task model

understand ready use tokenize dictionary fit need chapter cover following topic basic guideline control output tokenizer raw datum strategy preprocessing datum strategy word2vec tokenization problem limit create python program evaluate word2vec tokenizer build python program evaluate output byte level bpe algorithm customize nlp task specific vocabulary run unconditional conditional sample gpt-2 evaluate gpt-2 tokenizer step explore text text methodology define raffel et al 2019 match dataset tokenizer download benchmark dataset train transformer advantage datum prepare research lab use reference performance transformer model compare model datum need improve performance transformer furthermore imple- mente transformer model production require careful planning define good practice section define good practice avoid critical stumbling block example python cosine similarity measure limit tokenization encoding dataset let start good practice raffel et al 2019 define standard text text t5 transformer model go begin destroy myth raw datum preprocesse preprocesse datum reduce training time common crawl example contain unlabeled text obtain web extraction non text markup remove dataset google t5 team find text obtain common crawl reach level natural language english decide dataset need clean recommendation raffel et al 2019 apply corporate quality control good practice preprocessing quality control phase rule ap- ply example describe tremendous work require obtain acceptable real life figure 9.1 list key quality control process apply dataset figure 9.1 good practice transformer dataset match tokenizers datasets show figure 9.1 quality control divide preprocessing phase step 1 train transformer quality control transformer production step 2 let main aspect preprocessing phase step 1 preprocesse raffel et al 2019 recommend preprocesse dataset train model add extra idea transformer language learner teacher teach

machine student language explain proper english example need apply standard heuristic dataset sentence punctuation mark recommendation select sentence end punctuation mark period question mark remove bad word bad word remove list find follow site example tricky

word bad word remove list find follow site example tricky code content look generally good remove code content nlp task website contain page default lorem ipsum text necessary sure dataset content language wish excellent way start langdetect detect 50 + language https://pypi.org/project/ remove reference discrimination recommendation build knowledge base scrape web specific dataset hand suppress form discrimination certainly want machine ethical good idea run train transformer model dataset perform natural language inferences nli filter sentence sense bad information reference eliminate text refer link work unethical website person tough job certainly worthwhile list contain primary good practice require filter privacy law violation action specific project transformer train learn proper english need help detect problem input text production phase step 2 quality control train model behave like person learn language understand learn input datum input datum process step 1 preprocesse add new information training dataset training dataset turn knowledge base

corporate project user able run nlp task dataset obtain reliable answer question useful summary specific document apply good practice describe step 1

preprocesse real time input datum example transformer run input user nlp task summarize list document transformer powerful nlp model mean ethical responsibility heighten let good practice check input text real time accept bad information instead parse input real time filter unac- ceptable datum step 1 store reject datum reason filter user consult log display real time message transformer ask answer unfitting question match tokenizer dataset convert rare vocabulary standard vocabulary possible case 4 word2vec tokenization section chapter possible represent step forward stream datum transformer model analyze user input private

datum exclude dataset task authorize user country transformer run tricky topic consult legal adviser necessary go good practice let human quality control continuous human quality control transformer progressively complex nlp task human inter- vention remain mandatory think social medium giant automize discover content manager decide good bad platform right approach train transformer implement control output feed sig- nificant result training set training set continuously improve transformer continue learn figure 9.2 show continuous quality control help transformer training dataset grow increase performance production figure 9.2 continuous human quality control go good practice describe raffel et al 2019 add guidance base experience corporate ai project management let python program example limit encounter matching tokenizers datasets long thing think pretrained tokenizer like real life drive car year think engine day car break try find reason explain situation happen pretrained tokenizer result expect example word pair fit figure 9.3 figure 9.3 word pair tokenizer miscalculate example show figure 9.3 draw american declaration independence bill rights english magna carta cake chapter fit tokenizer compute have high value cosine similarity editor free ebook pay bill fit everyday english polysemy word meaning example bill mean pay refer bill rights result acceptable pure luck continue let moment clarify point qc refer quality control strategic corporate project qc mandatory quality

qc refer quality control strategic corporate project qc mandatory quality output determine survival critical project project strategic error acceptable strategic project error imply risk management audit intervention project continue abandon perspective quality control risk management tokenize dataset irrel- evant useless word critical word miss confuse embedding algorithm produce poor result chapter use word tokenizing loosely in- clude embedding impact strategic ai project poor result single error dramatic consequence espe- cially medical sphere airplane rocket assembly critical domain open tokenizer.ipynb base positional_encoding.ipynb

create chapter 2 getting start architecture transformer model result vary run stochastic nature word2vec algorithm prerequisite instal import pip install gensim==3.8.3 import numpy np nltk.tokenize import sent_tokenize word_tokenize gensim.models import word2vec import numpy np sklearn.metrics.pairwise import cosine_similarity import matplotlib.pyplot plt warnings.filterwarnings(action = ignore text.txt dataset contain american declaration independence bill rights magna carta work immanuel kant text tokenize text.txt train word2vec model @title word2vec tokenization sample = open("text.txt r s = sample.read process escape character matching tokenizers datasets f = s.replace("\n datum = sentence parsing sent_tokenize(f temp = tokenize sentence word j word_tokenize(i create skip gram model model2 = gensim.models word2vec(data min_count = 1 size = 512,window = 5 sg = 1 window = 5 interesting parameter limit distance current word predict word input sentence sg = 1 mean skip gram training algorithm output show size vocabulary 10816 dimensionality embedding 512 learning rate set alpha=0.025 word2vec(vocab=10816 size=512 alpha=0.025 word representation model embedding create cosine similarity function name similarity(word1,word2 send word1 word2 function return cosine similarity value high value high similarity function detect unknown word unk display message @title cosine similarity cosine = false default value keyerror keyerror exception raise print(word1 unk key find dictionary")#false keyerror keyerror exception raise cosine = false b true print(word2 unk key find dictionary cosine similarity calculate cosine==true mean word1 word2 know compute cosine similarity dot = np.dot(a b norma = np.linalg.norm(a normb = np.linalg.norm(b cos = dot norma normb aa = a.reshape(1,512 ba = b.reshape(1,512 cos_lib = cosine_similarity(aa ba function return cos_lib compute value cosine similarity case text.txt dataset let begin case 0 case 0 word dataset dictionary word freedom liberty dataset cosine similarity compute @title case 0 word text dictionary similarity limit 0.79 lot content insert text explore limit function similarity 0.79085565

freedom liberty match tokenizer dataset similarity algorithm iterative deterministic calculation

section result change dataset content dataset size run module version run cell 10 time obtain different value follow follow case obtain result 10 time google colab vm cpu run 1 similarity 0.62018466 freedom liberty run 2 similarity 0.62018466 freedom liberty run 10 similarity 0.62018466 freedom liberty factory reset runtime runtime menu google colab new vm cpu obtain run 1 similarity 0.51549244 freedom liberty run 2 similarity 0.51549244 freedom liberty run 10 similarity 0.51549244 freedom liberty perform factory reset runtime runtime menu google colab activate gpu new vm gpu obtain run 1 similarity 0.58365834 freedom liberty run 2 similarity 0.58365834 freedom liberty run 10 similarity 0.58365834 freedom liberty conclusion stochastic algorithm base probability good practice run prediction n time necessary let happen word miss case 1 word dataset dictionary miss word mean trouble way case send corporation right similarity function @title word(s case 1 word text dictionary dictionary contain word corporation corporation unk key find dictionary similarity 0

corporation right dead end word unknown unk token miss word provoke chain event problem distort transformer model output word important refer miss word unk possibility need check question answer unk dataset select tokenize dictionary unk dataset case word corporation explain dictionary case unk appear production user send input transformer contain token tokenize unk important word dataset usage transformer list problem continue grow transformer produce terrible result case consider 0.8 excellent performance transformer model specific down- stream task training phase real life want work system wrong 20 time nuclear plant maintenance team 0.8 satisfactory fuzzy environment like social media message lack proper language structure come bad suppose nlp team discover problem try solve chapter 4 pretraine roberta model scratch step 3 train tokenizer nightmare begin team use byte level bpe fix problem unk break word piece example end corporation corp + o + ra + tion + s. token high probability find dataset match tokenizers dataset unk set sub word represent token exist dataset convey original token meaning transformer train notice unk break piece train meaninglessly transformer produce excellent result performance 0.8 0.9

everybody applaud professional user apply erroneous result critical situation example english corp mean corporation corporal create confusion bad association corp word standard social medium use transformer trivial topic real life corporate project hard work produce pretrained tokenizer match dataset real life dataset grow day user input user input dataset model train update regularly example way ensure quality control following step train tokenizer byte level bpe algorithm control result program create control toke- nized datum section chapter train tokenizer word2vec algorithm quality control parse dataset find unk token store database run query check critical word miss unnecessary check process detail tempt rely transformer ability inference unseen word recommend run different quality control method strategic project critical decision making example legal summary law word difference lose win case court aerospace project airplane rocket 0 error tolerance standard quality control process run reliable transformer solution take lot legwork obtain reliable dataset paper write trans- former refer way work take produce acceptable dataset noisy relationship cause problem case 2 noisy relationship case dataset contain word etext declaration @title case 2 noisy relationship furthermore end tokenized dictionary similarity 0.880751 etext declaration well cosine similarity sure prediction exceed 0.5

stochastic nature algorithm produce different result run trivial social medium level look good professional level result disastrous etext refer project gutenberg preface ebook site explain matching dataset tokenizer section chapter goal transformer specific task understand editor preface understand content book depend usage transformer day sort example suppose editor want understand preface automatically use transformer generate preface text content declaration meaningful word relate actual content declaration independence etext preface project gutenberg add ebook produce erroneous natural language inference etext declaration transformer ask generate text let look miss word issue case 3 word text dictionary case word text dictionary distort result match tokenizer dataset let word pie logic @title case 3 word text

dictionary word pie dictionary pie unk key find dictionary similarity 0 pie logic assume word pie tokenized dictionary word word pie text file function pipeline detect word dictionary implement correction alternative function pipeline detect word dataset important let problem face rare word case 4 rare word rare word produce devaste effect output transformer specific task simple application manage rare word extend domain natural language example rare word occur dataset unnoticed model poorly train deal rare word medical legal engineering term professional jargon rare word slang hundred variation english language example different english word certain part united states united kingdom singapore india australia country rare word come text write century ago forget example case word justiciar @title case 4 rare word similarity judgement reasonable high similarity 0.6606605

justiciar judgement think word justiciar far fetched tokenizer extract magna carta date early 13 th century unfortunately program confuse obtain unexpected result run article magna carta valid 21 st century england example clause 1 13 39 40 valid famous magna carta following excerpt dataset 39 free man shall seize imprison strip right possession outlaw exile deprive standing way proceed force send lawful judgement equal law land 40 sell deny delay right justice implement transformer model law firm summarize document task careful let method use solve rare word problem case 5 replace rare word replace rare word represent project work reserve specific task project suppose corporate budget cover cost have knowledge base aeronautics example case worth spend necessary time query tokenize directory find word miss problem group topic solve knowledge base update regularly case 4 stumble word justiciar origin come french normand language root french latin like word judicaire note prediction vary run careful tokenizing embedding phase transformer matching tokenizer dataset replace word justiciar judge convey meta concept @title case 5 replace rare word produce interesting result need careful non deterministic aspect algorithm similarity 0.7962761 judge judgement word justiciar try word modern meaning compare judge try implement case 5 replace rare word case rare word need replace mainstream word result

satisfactory similarity 0.9659128 justiciar judge create query replacement word run find correlation 0.9 example manage critical legal project essential document contain rare word kind translate standard english transformer performance nlp task increase knowledge base corporation progressively increase let use cosine similarity entailment verification case 6 entailment case interested word dictionary test fix order example let pay + debt make sense similarity function @title case 6 entailment result satisfactory similarity 0.89891946

pay debt check dataset word pair check mean word pair extract email legal department example cosine similarity 0.9 email strip useless information content add knowledge base dataset company let pretrained tokenizer match nlp task standard nlp task specific vocabulary section focus case 4 rare word case 5 replace rare word word2vec tokenization section chapter use training_openai_gpt_2_ch09.ipynb rename version notebook train dataset chapter 7 rise suprahuman transformers gpt-3 engines change notebook dset dataset rename mdset contain medical content python function add control text tokenize byte level bpe describe training_openai_gpt_2_ch09.ipynb cover chapter 7 rise suprahuman transformers gpt-3 engines appendices iii iv

sure upload necessary file begin explain chapter 7 limit time wish train model interrupt order save model file github gpt-2 train_files directory chapter09 us- e notebook chapter 7 note dataset dset name mdset directory code let generate unconditional sample gpt-2 model train understand medical generate unconditional sample gpt-2 hand dirty section understand inner working transformer course skip chapter simply use openai api 4.0 ai specialist ai guru vaguely tell transformer model preprocessing pipeline order transformer model necessary understand transformer model work match tokenizer dataset case 4 rare word case 5 replace rare word see rare word word specific field old english variation english language world slang 2020 news fill medical term covid-19 outbreak section gpt-2 transformer cope medical text dataset encode train contain paper martina conte nadia loy 2020 name multi cue kinetic

model non local sensing cell migration fiber network chemotaxis title easy understand contain rare word load file locate gpt-2 train_files directory include mdset.txt run code explain chapter 7 rise suprahuman transformers gpt-3 engines run code cell cell chapter 7 guide special care follow instruction sure tf 1.x activate sure run step 4 restart runtime run step 4 tf 1.x cell continue error notebook get hand dirty use low level original gpt-2 code section api train model medical dataset reach unconditional sample cell step 11 generate unconditional sample @title step 11 generate unconditional sample import os import runtime restart python generate_unconditional_samples.py --model_name 117 m run cell stop wish produce random output community base machinery facilitate biofilm growth community member place biochemistry main discovery tool cell interact environment identify understand component effective mimicry 2 ol perception time take run command code notebook depend power machine notebook gpt-2 code explain educational purpose book recommend use openai api gpt-3 production response time fast transformer project cytic double truncation phase change ip

time fast transformer project cytic double truncation phase change ip polymerase call tcrec represent characteristic pattern double- cross enzyme alter fundamental configuration allow initiation maintenance process chop plainna vibrational operator soon radical modification occur translational parasubstitution tmt achieve uncontrolled activation syx trsi mutation introduce autophosphorylation tcmase sp important incorporate cellular double triad dtt signaling cell allow r h ofcourse ic 2a- > close look output notice following point structure generate sentence relatively acceptable grammar output bad non professional output

human like content make sense transformer unable produce real content relate medical paper train obtain well result hard work course increase size dataset contain look find wrong correlation datum example imagine medical project involve covid-19 dataset contain following sentence covid-19 dangerous virus like ordinary flu covid-19 dangerous virus covid-19 virus

create lab covid-19 certainly create lab vaccine dangerous vaccine lifesaver government manage pandemic correctly government necessary contradictory sentence discrepancy confirm dataset tokenizer customize specialized healthcare project aeronautic transporta- tion critical domain match tokenizer dataset imagine dataset billion word content conflictual noisy obtain reliable result matter try mean dataset small limit content scientific paper scientist disagree conclusion lot hard work solid team produce reliable result let try condition gpt-2 model generate train conditional sample section step 12 interactive context completion examples cell notebook run @title step 12 interactive context completion examples import os import runtime restart python interactive_conditional_samples.py --temperature 0.8 --top_k 40 --model_name 117 m --length 50 industry 4.0 ai specialist focus code transformer model model require level show unconditional datum tell vaguely condition gpt-2 model enter medical paper process cell sense environment respond external factor induce certain direction motion specific target taxi result persistent migration certain preferential direction guidance cue lead direct migration biochemical biophysical biochemical cue example soluble factor growth factor rise chemotaxis involve mono directional stimulus cue generate mono directional stimulus include instance bind ligand substratum induce haptotaxis durotaxi involve migration region increase stiffness ecm electrotaxis know galvanotaxis prescribe direct motion guide electric field current phototaxis refer movement orient stimulus light 34 important biophysical cue property extracellular matrix ecm alignment collagen fiber stiffness particular fiber alignment show stimulate contact guidance 22 21 tl;dr add tl;dr end input text tell gpt-2 model summarize text condition output make sense grammatically semantically ecm single tissue ecm effective address concern

develop novel imaging immunostaining scheme activate induce conversion protein output non deterministic response allow movement direct laser i.e. laser take pulse time laser direct target direct give direction nutshell mindful result well require research conclusion draw example chapter pretraine transformer mod- el vast amount random web crawl datum

example teach transformer english like transformer need train specific domain specialist field let investigation control tokenize datum control tokenize datum section read word gpt-2 model encode pretrained tokenizer run cell stop cell run subsequent additional tool control tokenize data cell training_openai gpt_2_ch09.ipynb notebook chapter cell add notebook chapter cell unzip out.npz contain encode medical paper dataset @title additional tool control tokenize datum zipfile zipfile('/content gpt-2 src out.npz r zip_ref match tokenizer datasets out.npz unzip read arr_0.npy numpy array contain encode dataset look load arr_0.npy contain encode dset import numpy np range(0,10 output element array 1212 5644 326 13 198 2682 open encoder.json convert python dictionary import encoder.json open("/content gpt-2 models/117m encoder.json r read_file print("converte json encode datum

python dictionary developer = json.load(read_file convert encode datum key value developer.items parse decode json datum print(key value finally display key value 500 token encode dataset search key value encode token range(0,500 key value developer.items print(key value word mdset.txt follow suggest add word sure gpt-2 pretrained tokenizer easily recognize case 1212 ġsuggest 5644 ġthat 326 easily recognize initial token precede initial whitespace character ġ how- let following word medical paper amoeboid rare word gpt-2 tokenizer break sub word ġam 716 o 78 eb 1765 oid 1868 let skip whitespace look happen amoeboid + o+ eb + oid agree unknown token unk byte level bpe strategy transformer attention layer associate sequence o sequence take apart contain o oid sequence contain oid possibly tabloid algorithm good news let following word amoeboid mesenchymal output clearly display rest token confusing ġam 716 o 78 eb 1765 oid 1868 ġand 290 ġmes 18842 match tokenizers datasets ench 24421 ym 4948 al 282 wonder problem reason sum word polysemy use word2vec tokenizer dictionary contain rare word amoeboid come unknown token use byte level bpe obtain overall well result exclude few variation word + ing token amoeboid bring polysemy problem low level sort prefix word + sub word + bush attention layer associate token create relationship exist define core problem polysemy nlu progress

need work hard improve nlp go lot everyday problem face real life project example time try example think useful leave use probe task verify level nlu transformer model explore scope gpt-3 powerful transformer openai gpt-3 limit let gpt- 3 react word amoeboid close medical term mainstream word need technical jargon project match dataset require quality control transformer organize dictionary embedding human detect error correct somebody example chapter explore word amoeboid control tokenized datum section chapter let ask gpt-3 amoeboid mean figure 9.4 ask gpt-3 amoeboid

let ask gpt-3 amoeboid mean figure 9.4 ask gpt-3 amoeboid mean amoeboid resemble amoeba adjective gpt-3 state noun output amoeboid noun mean resemble amoeba ask gpt-3 precise question obtain incorrect answer q amoeboid noun adjective amoeboid noun finally insist ask clear definition obtain correct answer q amoeboid mean medical term amoeboid mean resemble amoeba definition accurate grammatical analysis important real life project understand definition word identify role sentence adjective noun maybe grammatical aspect important educational grammar school project corporate supply chain finance e commerce application openai gpt-3 fine tune case see chapter 7 rise suprahuman transformer gpt-3 engine section conclude ensure datum need train trans- model tokenization process incomplete maybe take medical dictionary create large corpus medical article contain specific vocabulary model accurate tokenize dataset train model scratch 2022 developer development work think design lot let conclude chapter nlu task definition word sufficient medical project case gpt-3 sufficient definition sufficient srl prerequisite understand- e sentence match tokenizer dataset chapter measure impact tokenization subsequent datum encoding process transformer model transformer model attend token embedding positional encoding sub layer stack matter model encoder decoder encoder decoder model matter dataset good train tokenization process fail partly transformer model run miss see standard language task raw dataset train transformer discover pretrained tokenizer go billion word create dictionary small portion vocabulary come like tokenizer capture essence language learn remember important

word word frequently approach work standard task create problem specific task vocabulary look idea work limit standard tokenizer apply language checking method adapt text wish process tokenizer think encode datum apply method unconditional conditional task gpt-2 finally analyze limit datum tokenize match dataset gpt-3 lesson away chapter ai specialist stay time chapter semantic role labeling bert base transformer dig deep nlu use bert model ask transformer explain meaning sentence tokenized dictionary contain word exist language

transformer explain meaning sentence tokenized dictionary contain word exist language true false pretrained tokenizer encode dataset true false good practice check database true false good practice eliminate obscene datum dataset true false good practice delete datum contain discriminate assertion true false raw dataset produce relationship noisy content useful standard pretraine tokenizer contain english vocabulary past 700 year 8 old english create problem encode datum tokenizer train modern medical type jargon create problem encode datum to- kenizer train modern english true false 10 control output encode datum produce pretrained tokenizer good colin raffel noam shazeer adam roberts

katherine lee sharan narang michael matena yanqi zhou wei li peter j. liu 2019 explore limit transfer learning unified text text transformer https://arxiv.org/pdf/1910.10683.pdf openai gpt-2 github repository https://github.com/openai/gpt-2 n. shepperd github repository https://github.com/nshepperd/gpt-2 hugging face framework resource https://huggingface.co/ u.s. legal montana corporate law https://corporations.uslegal.com/state- martina conte nadia loy 2020 multi cue kinetic model non local sensing cell mi- gration fiber network chemotaxis https://arxiv.org/abs/2006.09707 declaration independence united states america thomas jefferson https:// united states bill rights united states related text https://www magna carta https://www.gutenberg.org/ebooks/10000 critique pure reason critique practical reason fundamental principles metaphysic moral https://www.gutenberg.org match tokenizers

dataset

join book discord space join book discord workspace monthly ask session author semantic role labeling transformer progress past year nlp past generation standard nlu approach learn syntactical lexical feature explain structure sentence nlp model train understand language basic syntax run semantic role labeling srl shi lin 2019 start paper ask preliminary syntactic lexical training skip bert base model perform srl go classical training phase answer yes shi lin 2019 suggest srl consider sequence labeling provide stan- dardize input format bert base model produce surprisingly good result chapter use pretrained bert base model provide allen institute ai base shi lin 2019 paper shi lin take srl level drop syntactic lexical training achieve begin define srl standardization sequence labeling input format start resource provide allen institute ai run srl task google colab notebook use online resource understand result finally challenge bert base model run srl sample sample srl work run difficult sample progressively push bert base model limit srl find limit model good way ensure real life implementation transformer model remain realistic pragmatic semantic role labeling bert base transformer chapter cover following topic define semantic role labeling define standardization input format srl main aspect bert base model architecture encoder stack manage mask srl input format bert base model srl attention process getting start resource provide allen institute ai build notebook run pretrained bert base model test sentence labeling basic example test srl difficult example explain result take bert base model limit srl explain step explore srl approach define shi lin 2019 getting start srl srl difficult human machine transformer take step close human baseline section define srl visualize example run pretrained let begin define problematic task srl define semantic role labeling shi lin 2019 advance prove idea find with- depend lexical syntactic feature chapter base peng shi jimmy lin research university waterloo

feature chapter base peng shi jimmy lin research university waterloo california show transformer learn language structure well attention layer srl label semantic role role word

group word play sentence rela- tionship establish predicate semantic role role noun noun phrase play relation main verb sentence example sentence marvin walk park marvin agent event occur- ring sentence agent doer event main verb govern verb walk predicate describe subject agent predicate provide information feature action subject approach refer predicate main verb example sentence marvin walk park predicate walk restricted form word park modify meaning walk modifier noun noun phrase revolve predicate argument argument term marvin example argument predicate walk srl require syntax tree lexical analysis let visualize srl example chapter allen institute visual code resource reference section information allen institute ai excellent interactive online tool represent srl visually chapter access tool allen institute ai advocate ai common good good use approach figure chapter create allennlp tool allen institute provide transformer model continuously evolve exam- ple chapter produce different result run good way chapter read understand concept explain merely run program time understand example provide run experiment sentence choice tool chapter visualize srl example figure 10.1 srl representation marvin walk park figure 10.1 srl representation sentence semantic role labeling bert base transformer observe follow label figure 10.1 verb predicate sentence argument argument sentence name arg0 modifier modifier sentence case location adverb adjective modify predicate meaning text output interesting contain short version label visual walked arg0 marvin v walked argm loc park define srl go example time look bert base model run pretrained bert base model section begin describe architecture bert base model chapter define method experiment srl sample bert model let begin look architecture bert base model architecture bert base model allennlp bert base model 12 layer encoder bert model allennlp team imple- mente

model 12 layer encoder bert model allennlp team imple- mente bert model describe shi lin 2019 additional linear classification description bert model minute necessary chapter 3 fine tuning bert models bert base model take advantage bidirectional attention simple approach architecture core potential transformer reside attention layer see transformer model encoder

decoder stack see transformer encoder layer decoder layer main advantage transformer remain near human approach attention layer input format predicate identification format define shi lin 2019 show far transformer come understand language standardized fashion cls marvin walk park.[sep walk sep training process standardize cls indicate classification exercise sep separator indicate end sentence sep follow predicate identification design author sep second separator indicate end predicate identifier format train bert model identify label semantic role let set environment run srl sample set bert srl environment google colab notebook allennlp visual text representation srl avail- able https://demo.allennlp.org/ semantic role labeling section apply following method open srl.ipynb install allennlp run sample 2 display raw output srl run visualize output allennlp online visualization tool display output allennlp online text visualization tool chapter self contain read run sample describe srl model output differ allennlp change transformer model allennlp model transformer general continuously train update dataset training change finally rule base algorithm produce result time output change run describe show screenshot let run srl experiment srl experiment bert base model run srl experiment method describe set bert srl environment section chapter begin basic sample sentence structure challenge bert base model difficult sample explore system capacity limit semantic role labeling bert base transformers open srl.ipynb run installation cell pip install allennlp==2.1.0 allennlp models==2.1.0 import tagging module train bert predictor allennlp.predictors.predictor import predictor predictor = predictor.from_path("https://storage.googleapis.com allennlp- add function display json object srl bert return display verb predicate description iterate json display excerpt prediciton prediction['verbs second display response include

iterate json display excerpt prediciton prediction['verbs second display response include tag print prediction print(json.dumps(prediction indent = 1 sort_keys = true time publication bert model specifically train semantic role labeling use model srl bert srl bert train ontonotes 5.0 dataset dataset contain sentence annotation dataset design identify predicate sentence

contain verb sentence identify word provide information verb verb come argument tell frame contain argument verb srl bert specialized model train perform specific task foundation model like openai gpt-3 see chapter 7 rise suprahuman transformer gpt-3 engines srl bert focus semantic role labeling acceptable accuracy long sentence contain predicate ready warm basic sample

basic sample intuitively simple tricky analyze compound sentence adjec- tive adverb modal difficult identify non expert human let begin easy sample transformer sample long relatively easy transformer bob think prepare meal 50 people hour run sample 1 cell srl.ipynb sentence="did bob think prepare meal 50 people hour bert srl identify predicate verb label result show excerpt head(prediction function verb v bob think prepare meal 50 people hour verb think arg0 bob argm adv v think arg1 prepare meal 50 people hour verb bob think v arg1 prepare meal 50 people hour verb prepare bob think arg0 argm mod v prepare arg1 meal

50 people argm tmp hour view response run full(prediction cell semantic role labeling bert base transformer use description argument dataset base structure propbank proposition bank verb think example excerpt interpret v identify verb think arg0 identify agent bob agent pro agent argm adv consider adverb adv argm mean adverb pro- vide adjunct necessary number run sample allennlp online interface obtain visual representation srl task frame verb verb figure 10.2 identify verb second verb identify think figure 10.3 identify verb think close look representation detect interesting property srl bert detect verb think avoid prepare trap interpret main verb instead prepare remain argument think detect adverb label verb figure 10.4 identify verb argument transformer move verb prepare label analyze context figure 10.5 identify verb prepare argument modifier simple bert base transformer model detect lot information grammatical structure sentence find verb prepare isolate noun label argument meal 50 people proto patient involve modification participant hour temporal modifier argm tmp modal modifier indicate modality verb like- lihood event analyze relatively long sentence

follow sentence easy contain verb mrs. mr. tomaso go europe vacation visit paris go visit eiffel tower semantic role labeling bert base transformer confusing sentence transformer hesitate let run sample 2 cell srl.ipynb notebook sentence="mrs mr. tomaso go europe vacation visit paris go visit eiffel tower excerpt output prove transformer correctly identify verb sentence verb go arg0 mrs. mr. tomaso v go arg4 europe argm- prp vacation visit paris go visit eiffel verb visit arg0 mrs. mr. tomaso go europe vacation v visit arg1 paris go visit eiffel tower verb go arg0 mrs. mr. tomaso go europe vacation visit paris argm tmp v go arg1 visit eiffel verb visit arg0 mrs. mr. tomaso go europe vacation visit paris go v visit arg1 eiffel tower run sample allennlp online

identify predicate generate frame frame go figure 10.6 identify verb go argument modifier interpret argument verb go mrs. mr. tomaso agent transformer find main modifier verb purpose trip europe result surprising know shi lin 2019 build simple bert model obtain high quality grammatical analysis notice go correctly associate europe transformer correctly identify verb visit relate paris figure 10.7 identify verb visit argument transformer associate verb visit directly eiffel tower stand ground right decision task ask transformer identify context second use verb go fall trap merge argument relate verb go twice sentence correctly split sequence produce excellent result figure 10.8 identify verb go argument modifier verb go twice transformer fall trap find temporal modifier verb go finally verb visit second time srl bert correctly interpret use figure 10.9 identify verb visit argument let run sentence bit confusing semantic role labeling bert base transformers sample 3 thing difficult transformer model follow sample contain variation verb drink time john want drink tea mary like drink coffee karim drink cool water faiza like drink tomato juice let run sample 3 srl.ipynb notebook sentence="john want drink tea mary like drink coffee karim drink cool water faiza like drink tomato juice transformer find way show follow excerpt output contain verb verb want arg0 john v wanted arg1 drink tea mary like drink coffee karim drink cool water faiza like drink tomato juice verb drink arg0 john want v drink arg1 tea mary like drink coffee karim drink cool

water faiza like drink tomato juice verb like john want drink tea arg0 mary v like arg1 drink coffee karim drink cool water faiza like drink tomato juice verb drink john want drink tea arg0

mary like v drink arg1 coffee karim drink cool water faiza like drink tomato juice verb drink john want drink tea mary like drink coffee arg0 karim v drank arg1 cool water faiza like drink tomato juice verb john want drink tea mary like drink coffee karim drink cool water arg0 faiza v like arg1 drink tomato juice verb like john want drink tea mary like drink coffee karim drink cool water arg0 faiza argm mod v like arg1 drink tomato juice verb drink john want drink tea mary like drink coffee karim drink cool water arg0 faiza like v drink arg1 tomato juice obtain visual representation run sentence allennlp online in- terface

examine perfect identify verb want make correct association figure 10.10 identify verb want argument identify verb drink correctly exclude faiza produce cool water argument figure 10.11 identify verb drink argument find bert base transformer produce relatively good result basic sample let try difficult one section run sample contain problem bert base transformer solve finally end intractable sample let start complex sample bert base transformer analyze semantic role labeling bert base transformer sample 4 take tricky srl territory sample separate alice verb like create long term dependency jump husband went jog sentence alice husband went jog sunday like dancing class meantime human isolate alice find predicate alice like dancing class meantime bert model find predicate like let find run code srl.ipynb sentence="alice husband went jog sunday like dancing class meantime output identify verb predicate label frame verb go alice arg0 husband v go arg1 jog argm- tmp sunday like dancing class meantime verb jog alice arg0 husband go v jog argm tmp sunday like dancing class meantime verb like arg0 alice husband go jog sunday v like arg1 dancing class meantime verb arg0 alice husband go jog sunday like v arg4 dancing class argm tmp meantime verb dance alice husband go jog sunday like v dancing class meantime let focus interested model find

predicate find verb like show excerpt output verb like separate alice predicate verb like arg0 alice husband go jog sunday let look visual representation model analysis run sample allennlp online ui transformer find alice husband figure 10.12 predicate go identify transformer explain predicate verb went husband argument jog argument relate go sunday temporal modifier represent raw output argm tmp figure 10.13 srl detection verb jogging verb jogging identify relate husband temporal modifier sunday semantic role labeling bert base transformer transformer stop detect alice like figure 10.14 identify verb like transformer detect analyze verb correctly figure 10.15 detect verb argument modifier temporal modifier meantime identify perfor- mance consider simple sequence + verb input srl bert train finally transformer identify verb dancing relate class figure 10.16 relate argument class verb dancing result produce sample 4 convincing let try find limit trans- sample 5 repeat verb time sample 5 contain word multi- ple function meaning go polysemy word round different meaning grammatical function word round noun adjective adverb transitive verb intransitive verb transitive intransitive verb round attain perfection completion sense round follow sentence use round past tense bright sun blue sky warm sand palm tree round verb round predicate sense bring perfection course accessible grammatical form rounded let happen let run sample 5 srl.ipynb sentence="the bright sun blue sky warm sand palm tree round output show verb transformer identify predicate fact find verb run full(prediction function online version interpret sentence well find verb figure 10.17 detect verb round argument like srl transformer kind verb form frequently let change sentence past tense present tense add s round bright sun blue sky warm sand palm tree round let srl.ipynb try present tense sentence="the bright sun blue sky warm sand palm tree round semantic role labeling bert base transformer raw output show predicate find show following

bert base transformer raw output show predicate find show following output

verb rounds arg1 bright sun blue sky warm sand palm tree v rounds argm prd run sentence

allennlp obtain visual explanation figure 10.18 detect word round verb bert base transformer word round find round present tense form bert model initially fail produce result expect little help friend end sample output vary evolution version model implement industry 4.0 pragmatic mindset require cognitive effort transformer let try sentence difficult label sample 6 take word think noun word suspect noun verb example ice verb hockey shoot puck way rink goal line opponent puck disk hockey hockey coach start day tell team train ice puck obtain imperative sentence coach yell ice puck guy note guy mean person regardless sex let run sample 6 cell happen sentence="now ice puck guy transformer fail find verb verb game transformer tremendous progress lot room developer improve model human game transformer online interface confuse puck verb figure 10.19 model incorrectly label puck verb problem solve model reach limit gpt-3 limit cope limit require expertise project success create specialized dictionary succeed project good news developer develop new cross disciplinary cognitive skill team appreciate try example sample srl approach limit explore develop preprocessing function transformer customize application leave let question motivation srl question scope srl face real life project job people satisfy ask project pragmatism come technical ideology 2020 ai ideology new ideology coexist end decade winner merge implement transformer customize application specialized jargon technical vocabulary reach intractable limit point semantic role labeling bert base transformer section question productivity srl motivation aspect limit predicate analysis question use term semantic limit predicate analysis srl rely predicate srl bert work long provide verb million sen- tence contain verb provide srl bert semantic role labeling section allennlp demo interface https://demo.allennlp.org/ assertion work happen

assertion answer question person 1 like drink person 2 cup coffee enter person 2 answer srl bert find figure 10.20 frame obtain output 0 total frame srl unable analyze sentence contain ellipsis predicate implicit explicit definition ellipsis act leave word sentence necessary understand hundred million sentence contain ellipsis speak write day srl bert yield 0 total

frame following answer question q begin yield 0 total q like breakfast

pancake hot chocolate model deduce pancake = proper noun = preposition hot= adjective q want london model deduce london = proper noun = adverb q work today model deduce subway = proper noun apply question middle dialogue obtain frame output srl bert context middle conversation person 2 want coffee q tea thank q ok hot chocolate q glass water see conversation frame semantic labeling find million example srl bert fail understand sentence contain predicate semantic role labeling bert base transformer finish movie concert exhibition review social medium produce 0 good movie bad concert life section show limit srl let redefine srl implement srl bert presuppose sentence contain predicate false assumption case analyze sentence base predicate analysis predicate contain verb predicate tell subject follow predicate contain verb additional information dog eat food quickly eat quickly tell way dog eat verb predicate problem reside fact verb predicate syntax grammar analysis semantic understand word fit grammatical functional point view restrictive sentence mean absolutely globydisshing maccaked tie srl bert perfectly perform semantic analysis sentence mean figure 10.21 analyze meaningless sentence draw conclusion example srl predicate analysis work verb sentence srl predicate analysis identify ellipsis predicate verb structure language grammatical analysis predicate analysis identify structure meaning sentence grammatical analysis go far predicate analysis necessary center semantics focus meaning phrase sentence semantic focus context way word relate grammatical analysis include syntax inflection function word phrase sentence term semantic role labeling mislead name predicate role labeling perfectly understand sentence predicate sequence structure sentiment analysis decode meaning sentence output predicate analysis sentiment analysis algorithm perfectly understand good movie positive regardless presence predicate recommendation use srl ai tool chapter 13 analyze fake news transformer let conclude exploration scope limit srl chapter explore srl srl task difficult human machine transform- er model show human baseline reach nlp topic certain

transform- er model show human baseline reach nlp topic certain extent find simple bert base transformer perform predicate sense disambiguation run simple transformer identify meaning verb predicate lexical syntactic labeling shi lin 2019 standard sentence + verb input format train srl analyze language restrictive srl ai pipeline ai tool productive add intelligence natural language semantic role labeling bert base transformer find transformer train strip sentence + predicate input solve simple complex problem limit reach relatively rare verb form limit final difficult problem add training dataset research team improve model discover ai good humanity exist allen institute ai free ai resource available addition research team add visual representation raw output nlp model help user understand ai see explain ai essential run program visual text representation provide clear view potential bert base model finally explore scope limit srl optimize use method ai tool transformer continue improve standardization nlp distribute architecture input format chapter chapter 11 let data talking story question answer challenge transformer task usually human perform explore potential transformer face named entity recognition ner question answer task semantic role labeling srl text generation task true false 2 predicate noun true false verb predicate true false 5 modifier adverb true false 6 modifier location true false bert base model contain encoder decoder stack true false 8

bert base srl model standard input format true false transformer solve srl task true false peng shi jimmy lin 2019 simple bert models relation extraction semantic role allen institute ai https://allennlp.org/ allen institute ai semantic role labeling resource https://demo.allennlp.org/ join book discord space join book discord workspace monthly ask session author let data talking story question answer reading comprehension require skill read text notice keyword main event create mental representation content answer question knowledge content representation examine question avoid trap make mistake matter powerful transformer answer open question easily open environment mean somebody ask question topic transformer answer correctly difficult possible extent gpt-3

chapter transformer use general domain training dataset closed question answer environment example critical answer medical care law inter- pretation require additional nlp functionality transformer answer question correctly regardless training environment close preprocessed question answer sequence transformer model wrong prediction sequence contain subject compound chapter focus method build question generator find unambiguous content text help nlp task question generator idea apply implement question answering begin show difficult ask random question expect transformer respond time let datum talking story question answer help distilbert model answer question introduce name entity recognition ner function suggest reasonable question addition lay ground question generator transformer add electra model pretraine discriminator question answering continue add semantic role labeling srl function blueprint text step section provide additional idea build reliable question answering solution include implement haystack framework finally straight gpt-3 davinci engine interface online explore question an- swering task open environment development training preparation end chapter build multi task nlp helper use cloud ai question answering chapter cover following topic limit random question answering ner create meaningful question base entity identification begin design blueprint question generator transformer test question find ner introduce electra encoder pretraine discriminator test electra model standard question srl create meaningful question base predicate identification project management guideline implement question answer transformer analyze create question generate srl output ner srl define blueprint question generator trans- explore haystack question answering framework roberta gpt-3 interface require development preparation let begin go methodology apply analyze generation ques- tion question answer task question answer mainly present nlp exercise involve transformer dataset contain ready ask question answer question transformer train answer question ask closed environment complex situation reliable transformer model implementation require cus- transformers method perfect efficient universal transformer model question answer nlp task exist good model project produce good

output specific dataset task method outperform model case example suitable

output specific dataset task method outperform model case example suitable method average model produce efficient result flawed method excellent model chapter run distilbert electra roberta model produce well performance performance guarantee result critical domain example space rocket spacecraft production project ask nlp bot question mean obtain exact answer suppose user need ask question page report status regener- atively cool nozzle combustion chamber rocket question specific cooling status reliable line information user want nlp bot cut long story short let nlp bot transformer model literal statistical answer quality cognitive control risky happen trustworthy nlp bot connect knowledge base contain datum rule run rule base expert system background check nlp bot answer nlp transformer model bot produce smooth reliable natural language answer possibly human voice universal transformer model method fit need exist project re- quire specific function customize approach vary tremendously depend user expectation let datum talking story question answer chapter focus general constraint question answering specific trans- model choice chapter question answer project guide introduction transformer question answering focus question answering open environment question prepare transformer model require help nlp task classical program explore method idea combine task reach goal project method 0 explore trial error approach ask question randomly method 1 introduce ner help prepare question answer task method 2 try help default transformer electra transformer model introduce srl help transformer prepare question introduction method show single question answer method work high profile corporate project add ner srl improve linguistic intelligence transformer agent solution example ai nlp project implement question answering defense project tactical situation aerospace corporation combine different nlp method ensure answer provide 100 reliable design multi method solution project implement let start trial error approach method 0 trial error question answering easy true let find open qa.ipynb google colab notebook chapter run notebook cell cell run cell install

colab notebook chapter run notebook cell cell run cell install hugging face transformer framework implement chapter pip install -q transformer note hugging face transformer continually evolve update library module adapt market default version work pin pip install transformers==[version run function notebook import hugging face pipeline contain ready use transformer resource provide high level abstraction function hugging face library resource perform wide range task access nlp task simple api pro- gram create google colab recommend run google colab vm free pipeline import line code transformer import pipeline line option instantiate transformer model task perform nlp task default model default tokenizer perform nlp task custom model perform nlp task custom model custom tokenizer pipeline('<taskname > model='<model > tokenizer='<tokenizer let begin default model tokenizer nlp_qa = pipeline('question answer provide text use submit question transformer sequence = traffic begin slow pioneer boulevard los angeles make difficult city wbgo play cool jazz weather cool make pleasant make city friday afternoon

nat king cole singe jo maria slowly way la drive barstow plan las vegas early evening nice dinner sequence deceptively simple need plug line code api ask question obtain answer nlp_qa(context = sequence question='where pioneer boulevard output perfect answer answer los angeles end 66 score 0.988201259751591

start let data talking story question answer implement question answer transformer nlp task line code download ready use dataset contain text question answer fact chapter end right set question answer task thing simple real life implementation suppose implement question answer transformer model user ask question document store database significant constraint need run transformer set key document create question system work guarantee transformer answer question correctly question immediately arise go find question ask test system expert agree job happen question produce train model result satisfactory happen question answer

matter model use train work limited sample process take long scale cost try question come expert help one work one forever trial error solution chapter aim provide method tool reduce cost implement question answer transformer model find good question question answering challenge implement new dataset customer think transformer lego ® set building block assemble fit encoder decoder stack use set small large extra large xl think nlp task explore book lego ® set solution project implement assemble nlp task reach goal like software implementation trial error search question methodical approach chapter continue run qa.ipynb cell cell explore method describe use allennlp ner interface obtain visual representation ner srl result enter sentence interface go https://demo allennlp.org/reading-comprehension select named entity recognition se- mantic role labeling enter sequence chapter allennlp model account want obtain visual representation let start try find right xl transformer model question question answer ner method method 1 ner section use ner help find idea good question transformer model con- tinuously train update dataset training change finally rule base algorithm produce result time output change run ner detect people location organization entity sequence run ner task main part paragraph focus ask question ner find question continue run qa.ipynb cell cell program initialize pipeline ner task perform default model tokenizer nlp_ner = pipeline("ner continue

ner task perform default model tokenizer nlp_ner = pipeline("ner continue use deceptively simple sequence run method 0 trial error section chapter sequence = traffic begin slow pioneer boulevard los angeles make difficult city wbgo play cool jazz weather cool make pleasant make city friday afternoon nat king cole singe jo maria slowly way la drive barstow plan las vegas early evening nice dinner run nlp_ner

cell qa.ipynb let datum talking story question answer output generate result nlp task score round decimal place fit width page word pioneer score 0.97 entity loc index 8 word boulevard score 0.99 entity loc index 9 word los score 0.99 entity loc index 11 word angeles score 0.99

entity loc index 12 word w score 0.99 entity org index 26 word b score 0.99 entity org index 27 word g score 0.98 entity org index 28 word o score 0.97 entity org index 29 word nat score 0.99 entity index 59 word king score 0.99 entity index 60 word cole score 0.99 entity index 61 word jo score 0.99 entity index 65 word maria score 0.99 entity index 67 word la score 0.99 entity loc index 74 word bar score 0.99 entity loc index 78 word sto score 0.85 entity loc index 79 word w score 0.99 entity loc index 80 word las score 0.99 entity loc index 87 word vegas score 0.9989519715309143 entity loc index documentation hugging face describe label case main one person org organization loc location result correct note barstow split token let run sequence allennlp name entity recognition section https://demo allennlp.org/named-entity-recognition obtain visual representation sequence figure 11.1 ner ner highlight key entity use create question ques- let ask transformer type question question relate location question related person let begin

location question location entity question qa.ipynb produce nearly 20 entity location entity particularly interesting word pioneer score 0.97 entity loc index 8 word boulevard score 0.99 entity loc index 9 word los score 0.99 entity loc index 11 word angeles score 0.99 entity loc index 12 word la score 0.99 entity loc index 74 let data talking story questions answer word bar score 0.99 entity loc index 78 word sto score 0.85 entity loc index 79 word w score 0.99 entity loc index 80 word las score 0.99 entity loc index 87 word vegas score 0.9989519715309143 entity loc index apply heuristic method create question output qa.ipynb generate merge location original form parser apply template location scope book write classical code project write function work show pseudocode range beginning output end output filter record contain loc merge loc fit save merge loc question answer ner output loc pioneer boulevard loc los angeles loc las vegas generate question automatically template example apply random function write function job show following pseudocode location location template 1 loc template 2 loc locate obtain question automatically example pioneer boulevard los angeles locate la barstow las vegas locate know question directly answer

sequence create manage automatically suppose question create automatically method enter sequence create question automatically let suppose question create automatically let run nlp_qa = pipeline('question answer print("question 1.",nlp_qa(context = sequence question='where pioneer print("question 2.",nlp_qa(context = sequence question='where los print("question 3.",nlp_qa(context = sequence question='where la print("question 4.",nlp_qa(context = sequence question='where barstow print("question 5.",nlp_qa(context = sequence question='where las vegas output show question 1 answer correctly question 1 score 0.9879662851935791 start 55 end 67 answer los angeles question 2 score 0.9875189033668121 start 34 end 51 answer pioneer boulevard question 3 score 0.5090435442006118 start 55 end 67 answer los angeles question 4 score 0.3695214621538554 start 387 end 396 answer las

question 4 score 0.3695214621538554 start 387 end 396 answer las vegas question 5 score 0.21833994202792262 start 355 end 363 let data talking story question answer output display score start end position answer answer score question 2 0.98 run wrongly state los angeles time control transformer project management add quality decision making examine example manage transformer hard code function manage automatically classify project manage- ment example project level easy intermediate difficult difficult project management scope book briefly category easy project website elementary school teacher delight see text display html page answer question obtain automatically merge development assertion fixed format loc loc example barstow california add true false assertion teacher administrator interface allow teacher click right answer finalize multiple choice questionnaire 2 intermediate project encapsulate transformer automatic question answer program use api check answer correct auto- matically user process seamless wrong answer transformer store analysis difficult project implement intermediate project chatbot fol- low question example transformer correctly place pioneer boulevard los angeles chatbot user ask natural follow question near la require development difficult project research project train transformer recognize loc entity

million record dataset output result real time streaming map software api good news find way use find bad news implement transformer ai real life project require powerful machine tremendous teamwork project manager subject matter experts smes developer end user let try person entity question person entity question let start easy question transformer nlp_qa = pipeline('question answer nlp_qa(context = sequence question='who singe answer correct state sequence singe answer nat king cole ask transformer question require thinking clearly nlp_qa(context = sequence question='who go las vegas impossible answer question take sentence apart transformer make big mistake answer nat king cole transformer honest display score 0.35 score vary calculation transformer model transformer face semantic labeling problem let try well person entity question apply srl method

problem let try well person entity question apply srl method method 2 srl transformer find drive las vegas think nat king cole instead jo maria go wrong transformer think obtain explanation find let semantic role modeling necessary minute review chapter 10 semantic role labeling bert base transformer let data talking story question answer let run sequence allennlp semantic role labeling section https://demo allennlp.org/semantic-role-labeling

obtain visual representation verb drive sequence run srl bert model previous chapter figure 11.2 srl run text srl bert find 19 frame section focus drive problem argument verb drive jo maria inference true let ask question qa.ipynb nlp_qa(context = sequence question='who note result vary run allennlp update mind transformer model evolve output vary how- concept remain output correct answer jo maria find way ask question obtain right answer try paraphrase nlp_qa(context = sequence question='who drive las vegas obtain somewhat well result answer nat king cole singe jo maria transformer understand nat king cole singe jo maria need find way ask well question let try model question answer electra switch model need know output show model distilbert model train question answering model 6 layer 768 feature show layer 6 layer number 0 n dropout dropout(p=0.1 inplace = false q_lin

linear(in_features=768 out_features=768 bias = true k_lin linear(in_features=768 out_features=768 bias = true v_lin linear(in_features=768 out_features=768 bias = true let data talking story questions answer out_lin linear(in_features=768 out_features=768 try electra transformer model clark et al 2020 design transformer model improve masked language modeling mlm pretraining method chapter 3 fine tune bert models masked language modeling subsection see bert model insert random mask token mask training process

clark et al 2020 introduce plausible alternative generator network simply random token bert model train predict identity mask corrupt token clark et al 2020 train electra model discriminator predict mask token generate token figure 11.3 show electra train figure 11.3 electra train discriminator figure 11.3 show original sequence mask go generator generator insert acceptable token random token electra transformer model train predict token come original sequence replace architecture electra transformer model hyperparameter bert transformer model want obtain well result cell run qa.ipynb ques- tion answer cell electra small generator nlp_qa = pipeline('question answer model='google electra small-nlp_qa(context = sequence question='who drive las vegas output expect answer slow pioneer boulevard los angeles make output change run transformer model idea remain output send training message expect initialize electraforquestionanswering checkpoint model train task expect initialize electraforquestionanswering checkpoint model expect exactly identical like warning message conclude bad model explore avenue offer electra require training course experiment possible find new idea decide train model think step project management constraint obtain result expect default distilbert electra trans- main option solution train distilbert electra model additional dataset train data- set costly process real life project training month new dataset need implement hyperparameter change hardware cost need take account furthermore result unsatisfactory project manager shut project try ready use transformer fit need hugging face model https://huggingface.co/transformers/usage find way obtain well result additional nlp task help question an- chapter focus find additional nlp task

help default distilbert model let use srl extract predicate argument let datum talking story question answer srl find question allennlp use bert base model implement srl.ipynb notebook chapter 10 semantic role labeling bert base transformers let rerun sequence allennlp semantic role labeling section https://demo allennlp.org/semantic-role-labeling obtain visual representation predicate enter sequence work traffic begin slow pioneer boulevard los angeles make difficult city wbgo play cool

pioneer boulevard los angeles make difficult city wbgo play cool jazz weather cool make pleasant make city friday afternoon nat king cole singe jo maria slowly way la drive barstow plan las vegas early evening nice dinner bert base model find predicate goal find property srl output automatically generate question base verb sentence list predicate candidate produce bert model verbs={"began slow making"(1 play making"(2 making"(3 singe drove plan write program start introduce verb counter show fol- range verb verb verb counter >

max_count filter verb counter exceed number acceptable occurrence max_count verb ex- clude experiment development difficult disambiguate multiple semantic role verb argument let past tense list list limit verbs={"began slow play singing drove plan continue write function filter verb look verb lengthy argu- ment verb begin long argument figure 11.4 srl apply verb began argument begin long fit screenshot text version show difficult interpret argument begin begin traffic v begin arg1 slow pioneer boulevard los angeles make difficult city wbgo play cool jazz weather cool make pleasant make city friday afternoon nat king cole singe jo maria slowly way la drive barstow plan las vegas early evening nice dinner add function filter verb contain argument exceed maximum length range verb verb verb length(argument verb)>max_length filter verb length verb argument exceed maximum length max_length verb exclude experiment moment let begin list list limit verbs= slow playing singe drove plan let data talking story question answer add exclusion rule depend project work maxlength function restrictive max_length value extract potentially interesting candidate automatic

question generator verb candidate short- est argument transform question verb slow fit rule set appear sequence argument long contain short argument sequence allennlp visual representation confirm choice figure 11.5 srl apply verb slow text output easily parse slow arg1 traffic begin v slow arg1 argm adv pioneer boulevard argm loc los angeles argm adv make difficult city result follow output vary evolve transformer model idea remain verb slow identify key aspect srl output automatically generate template generate template argument label person write function

manage possibility show following pseudocode ner(argi)==i template = verb ner(argi)!=i template = verb function require work deal verb form modifier experiment apply function generate following question slow let run default pipeline following cell nlp_qa = pipeline question answer nlp_qa(context= sequence question='what slow result satisfactory answer traffic default model case distilbert correctly answer question automatic question generator following run ner automatically parse result classical code generate entity question run srl automatically filter result rule generate srl question ner result determine template use solution means complete work need probably require ad- ditional nlp task code give idea hard work implement ai let try approach filter verb playing visual representation show argument wbgo cool jazz figure 11.6 srl apply verb playing let data talking story question answers text version easy parse playing traffic begin slow pioneer boulevard los angeles make difficult city argm dis arg0 wbgo v play arg1 cool jazz run whowhat function argument template choose template following question generate automatically play let run default pipeline question following cell

nlp_qa = pipeline('question answer nlp_qa(context = sequence question='what play output satisfactory answer cool jazz singing good candidate whowhat function find template auto- matically generate follow question singe successfully test question chapter verb drive tag problem transformer solve problem verb good candidate result following output vary evolve

transformer model idea remain identify verb argument figure 11.7 srl apply verb additional development produce template correct verb form let suppose work ask model following question nlp_qa = pipeline('question answer nlp_qa(context = sequence question='who see output wrong argument answer nat king cole presence nat king cole jo maria sequence complex sequence create disambiguation problem transformer model nlp model project management research require easy way implement question answering shortcut begin implement method generate question automatically automatic question generation critical aspect nlp transformer model need pretraine multi task dataset contain ner srl question answering problem solve project manager need learn combine nlp task help solve specific task question answering let datum talking story question answer coreference resolution https://demo.allennlp.org/coreference-resolution help model identify main subject sequence work result produce allennlp show interesting analysis figure 11.8 coreference resolution sequence continue develop program add output coreference resolution set0={'los angeles city la set1=[jo maria

add coreference resolution pretraine task add post process task question generator case question generator simulate human behavior consid- erably enhance performance question answer task include customize additional nlp task pretraine process question answer model course decide use new strategy pretrain model run chapter distilbert electra let user ask question wish recommend work question generator question answer task question educational purpose train transformer provide idea real time user work pretraine transformer model include specific nlp task im- prove question answer performance use question generator train explore haystack roberta model haystack question answer framework interesting functionality worth explore fit need give project section run question answer sentence experiment model method chapter cell install module necessary run haystack install haystack pip install farm haystack==0.6.0 install specific version urllib torch avoid conflict preinstalle version colab pip install urllib3==1.25.4 pip install torch==1.6.0+cu101 f

https://download.pytorch.org/whl/torch notebook use roberta model load local model qa model hugging face model hub haystack.reader.farm import farmreader reader = farmreader(model_name_or_path="deepset roberta base squad2 use gpu = true no_ans_boost=0 return_no_answer = false chapter 4 pretraine roberta model scratch general description roberta model remain cell notebook answer question text explore detail chapter text = traffic begin slow pioneer boulevard nice dinner compare answer obtain previous section output decide transformer model like implement explore q&a gtp-3 engine section try avoid training fine tuning load program server dataset instead user simply connect openai account use interactive gpt-3 engine online educational interface provide sufficiently good answer provide e explanation t text follow let data talking story question answer e = answer question text t = traffic begin slow pioneer boulevard nice dinner question ask answer obtain form question answer go las vegas jo maria singe nat king cole kind music play jazz plan evening nice dinner need run wide range educational nlp task online interactive interface

api gpt-3 engine change s show gpt-3 expect e create endless interaction generation nlp bear industry 4.0 developer consultant project manager need acquire new set skill cognitive approach linguistic psychology cross dis- ciplinary dimension necessary time chapter 7 rise suprahuman transformers gpt-3 engine explore critical aspect use question answer transformer let sum work chapter find question answering easy implement transformer model take minute get work hour month ask default transformer hugging face pipeline answer simple ques- tion distilbert default transformer answer simple question choose easy question real life user ask kind question transformer confuse produce erroneous output decide continue ask random question random answer begin design blueprint question generator productive solution start ner find useful content design function automatically create question base ner output quality promising require work try electra model produce result expect stop minute decide spend costly resource train transformer model design add srl blueprint question generator test question produce add ner analysis generate meaningful

question haystack framework introduce discover way address question answering finally run example gpt-3 engine directly openai educational interactive interface api cloud ai platform increase power accessibility experiment lead conclusion multi task transformer provide well performance complex nlp task transformer train specific task implement transformer require prepare multi task training heuristic classical code question generator question generator train model question training input datum standalone solution chapter detect customer emotion prediction explore im- plement sentiment analysis social medium feedback train transformer model answer question true false question answering require research perfect true false name entity recognition ner provide useful information look mean- ingful question true false semantic role labeling srl useless prepare question true false 5 question generator excellent way produce question true false implement question answering require careful project management true false electra model architecture gpt-2 true false 8

electra model architecture bert train discriminator ner recognize location label loc true false 10 ner recognize person label person true false let data talking story question answer allen institute ai https://allennlp.org/ allen institute reading comprehension resource https://demo.allennlp.org/ kevin clark minh thang luong quoc v. le christopher d. manning 2020 electra pretraine text encoder discriminator generator https://arxiv.org/ hug face pipeline https://huggingface.co/transformers/main_classes/ github haystack framework repository https://github.com/deepset-ai/haystack/

join book discord space join book discord workspace monthly ask session author detect customer emotion prediction sentiment analysis rely principle compositionality understand sentence understand part sentence tough task possible nlp trans- model try transformer model chapter find start stanford sentiment treebank sst sst provide dataset com- plex sentence analyze easy analyze sentence movie great happen task tough complex sentence movie bit long enjoy sentence segment force transformer model understand structure

sequence logical form test transformer model complex sentence simple sentence find matter model try work train transformer model like student need work hard learn try reach real life run distilbert roberta large bert base minilm l12 h84 uncase bert base multilingual model fun discover student require training like way use output sentiment task improve customer relationship nice star interface implement website finally use gpt-3 online interface sentiment analysis openai account ai development api require detect customer emotion prediction chapter cover following topic sst sentiment analysis define compositionality long sequence sentiment analysis allennlp roberta run complex sentence explore new frontier transformer hug face sentiment analysis model distilbert sentiment analysis experiment minilm l12 h384 uncase look bert base multilingual model sentiment analysis gpt-3 let begin go sst getting start sentiment analysis transformer section explore sst transformer use train model sentiment use allennlp run roberta large transformer stanford sentiment treebank sst socher et al 2013 design semantic word space long phrase define principle compositionality apply long sequence principle compositionality mean nlp model examine constituent expression complex sentence rule combine understand meaning sequence let sample sst grasp meaning principle compositionality section chapter self contain choose perform action describe read chapter view screenshot provide interactive sentiment treebank https://nlp.stanford.edu/sentiment/treebank selection

wish graph sentiment tree appear page click image obtain sentiment tree figure 12.1 graph sentiment tree example click graph number 6 contain sentence mention jacques derrida pioneer deconstruction theory linguistic long complex sen- enlighten derrida lecture self derrida undeniably fascinating playful fellow socher et al 2013 work compositionality vector space logic form example define rule logic govern jacques derrida sample imply un- derstanding following word comma separate phrase rest sentence interpret understand second sentence comma vector space define socher et al 2013 produce complex graph represent principle compositionality detect customer emotion prediction view graph section

section section segment figure 12.2 segment complex sentence sentence correctly split main part second segment correct figure 12.3 main segment complex sentence detect customer emotion prediction draw conclusion method socher et al 2013 design sentiment analysis reduce count positive negative word sen- transformer model nlp model able learn principle compo- sitionality understand constituent complex sentence fit logical form rule transformer model able build vector space interpret subtilitie theory practice roberta large model sentiment analysis roberta large section use allennlp resource run roberta large transformer liu et al 2019 analyze exist bert model find train expect consider speed model produce surprising work improve pretraining bert model produce robustly optimize bert pretraining approach roberta let run roberta large model sentimentanalysis.ipynb run cell install allennlp model pip install allennlp==1.0.0 allennlp models==1.0.0 let try run jacques derrida sample echo sentence enlighten derrida lecture self derrida undeniably fascinating playful fellow |

allennlp predict https://storage.googleapis.com/allennlp-public-models/ output display architecture roberta large model 24 layer 16 attention head necessary minute description bert architecture bert model configuration section chapter 3 fine tuning bert models advantage model sentiment analysis produce value 0 negative 1 positive output produce result sentiment analysis task display output logit final positive result prediction logit 3.646597385406494 -2.9539334774017334 prob output contain token id vary run final token_id 0 5994 50 45 47 769 38853 30 143 9 6113 10505 281 25798 15 5 97 8 5 1403 2156 211 14385 4347 16 41 35559 12509 8 23317 2598 479 2 label 1 note algorithm stochastic ouput vary run detect customer emotion prediction output display token token < s > \u0120whether \u0120or \u0120not \u0120you \u0120re \u0120enlightened \u0120by \u0120any \u0120of u0120der rid \u0120lectures \u0120on \u0120the u0120other \u0120and \u0120the \u0120self \u0120 u0120d err ida \u0120is \u0120an \u0120undeniably u0120fascinating \u0120and \u0120playful \u0120fellow time enter sample explore design pretraine roberta let use sentiment analysis predict customer behavior trans- predict customer behavior sentiment

analysis section run sentiment analysis task hugging face transformer model one produce good result one simply like well begin hugging face distilbert model sentiment analysis distilbert let run sentiment analysis task distilbert use result predict open sentimentanalysis.ipynb transformer installation import cell pip install -q transformer transformer import pipeline create function name classify run model sequence send nlp_cls = pipeline('sentiment analysis note send m=1 function display configuration distilbert 6 layer 12 head model specific parameter distilbert model label definition

detect customer emotion prediction create list sequence add send classify function sequence="the battery model9x phone 6 hour unhappy sequence="the battery model9x phone 6 hour unhappy mad buy moel10x thing well super satisfied sequence="the customer unhappy sequence="the customer satisfied m=0 display model cofiguration=1 default=0 case seq=3 activate simulate customer issue need account output negative example look label negative score 0.9997098445892334 draw conclusion result predict customer behavior write func- tion store prediction customer management database count number time customer complain service product period week month year customer complain switch competitor well product service detect product service occur negative feedback message product service faulty require quality control improvement minute run sequence create sequence explore dis- explore hugging face transformer sentiment analysis hugging face model list section explore hugging face transformer model list enter sample evaluate result idea test model model fit need well give project run hugging face model https://huggingface.co/models model use find description model documentation provide hugging face https://huggingface.co/transformers/. test model implement find require fine tuning pretraining nlp task wish perform case hugging face trans- former follow fine tuning refer chapter 3 fine tune bert models pretraining refer chapter 4 pretraine roberta model scratch let list hugging face model https://huggingface.co/models select text classification task pane figure 12.4 select text classification model detect customer emotion prediction list transformer model train text

classification appear figure 12.5 hugging face pretraine text classification model default sort mode sort download search exciting transformer model test online begin distilbert distilbert sst distilbert base uncase finetune sst-2 english model fine tune sst let try example require good understanding principle compositionality customer unhappy fact satisfied think time give false impression sentence tough transformer analyze require logical rule training output false negative figure 12.6 output complex sequence classification task false negative mean model work correctly choose model mean download train long

mean model work correctly choose model mean download train long well time write book bert like model good ranking glue superglue leaderboard ranking continuously change fundamental con- cept transformer try difficult complicated example example crucial lesson real life project try estimate time customer complain false negative false positive regular human intervention mandatory year let minilm model try detect customer emotion prediction microsoft minilm l12 h384 uncase optimize size self attention layer teacher tweaking bert model obtain well performance 12 layer 12 head 33 million parameter 2.7 time fast bert base let test capacity understand principle compositionality customer unhappy fact satisfied think time give false impression output interesting produce careful split undecided score figure 12.7 complex sentence sentiment analysis output conclusive 0.5 positive let try model involve entailment multi genre natural language inference multinli task https://cims.nyu.edu/~sbowman/ multinli/ help solve interpretation complex sentence try deter- customer mean inference task determine sequence entail follow need format input split sequence sequence splitting token customer unhappy</s></s >

fact satisfied think time give false impression result interesting remain neutral figure 12.8 neutral result obtain slightly positive sentence mistake result second sequence infer sequence result carefully correct let finish experiment positive sentiment multilingual bert base model detect customer emotion prediction bert base multilingual model let run final experiment

super cool bert base model nlptown bert base- design let run friendly positive sentence english figure 12.9 sentiment analysis

english let try french ce modèle est super bien model super good figure 12.10 sentiment analysis french path model hugging face nlptown bert base multilingual uncased- sentiment find search form hugging face website present link detect customer emotion prediction implement website following initialization code transformer import autotokenizer automodelforsequenceclassification tokenizer = autotokenizer.from_pretrained("nlptown bert base multilingual- model = automodelforsequenceclassification.from_pretrained("nlptown bert- time patience result super cool implement transformer website average global satisfaction customer use continuous feedback improve customer service anticipate customer reaction leave gpt-3 perform sentiment analysis sentiment analysis gpt-3 need openai account run example section educational interface require api development training simply enter tweet example ask sentiment analysis tweet find movie exciting enjoy watch tweet eat spicy food like find super good output satisfactory submit difficult sequence gpt-3 engine tweet difficult find enjoy life parameter account output false sentiment positive sentence show difficulty life word enjoy introduce bias gpt-3 enjoy sequence replace verb output negative tweet difficult find life parameter account output false life difficult figure conclude sentence negative correct output neutral ask gpt-3 perform task explain difficult pipeline example run nlp task user show industry 4.0 i4.0 go human intervention automatic functionality know situation require new skill design preprocessing function transformer produce expect result human useful example tweet classification ready use code describe run openai gpt-3 task section chapter 7 rise suprahuman transformer gpt-3 engine implement example section code wish let prove valuable asset pragmatic i4.0 thinking leave sentiment analysis hug face transformer contain sentence come true label sentence neutral bother curious openai gpt-3 well gpt-3 foundation model theoretically thing train examine sentence customer unhappy fact satisfied think time give false impression read sentence closely customer look deeply

understand fact satisfied decide try model blindly reach work try model productive detect customer emotion prediction need root problem logic experimentation want rely

customer emotion prediction need root problem logic experimentation want rely algorithm find reason automatically need use neuron problem difficult identify customer machine chapter 10 semantic role labeling bert base transformer let ask srl bert investigate srl run satisfied semantic role labeling interface https://demo result correct figure 12.11 srl simple sentence analysis clear frame predicate verb arg1 satisfied find analysis complex sentence figure 12.12 verb satisfied merge word cause confusion satisfied arg2 problem focus argm adv modify word false misleading argm adv relative arg2 contain thinking thinking predicate give false impression thinking identify predicate complex sentence unidentified ellipsis see question scope srl section chapter 10 quickly verify enter sentence ellipsis customer unhappy fact satisfied think time give false impression problem srl ellipsis see chapter 10 correct predicate accurate frame frame 1 show unhappy correctly relate figure 12.13 unhappy correctly relate frame 2 show satisfied separate sentence individually identify argument complex sentence figure 12.14 satisfied separate word arg2 let straight predicate contain thinking verb want bert srl analyze correctly suppress ellipsis repeat sentence output correct figure 12.15 accurate output ellipsis detect customer emotion prediction leave srl investigation clue word false confusing argument algorithm relate word ellipsis repetition gpt-3 let hugging face clue investigate hugging face let distilbert base uncased fine tune sst-2 model chapter distilbert sst section investigate clue ellipsis submit sentence ellipsi customer unhappy fact satisfied think time give false impression output remain negative figure 12.16 false negative presence false positive sentence false sentence leave ellipsi customer unhappy fact satisfied think time give impression bingo output positive figure 12.17 true positive know word false create confusion srl ellipsis thinking know false create confusion sentiment analysis hugging face transformer model gpt-3 well let detect customer emotion prediction investigate gpt-3 playground let use openai example advanced tweet classifier modify satisfy in-

vestigation step

openai example advanced tweet classifier modify satisfy in- vestigation step step 1 show gpt-3 expect sentence customer satisfied sentence customer satisfied sentence service sentence link review step 2 show example output format expect 1 love new batman movie 2 hate phone battery die 3 day 4 link article 5 new music video blow mind sentence sentiment rating step 3 enter sentence number 3 1 stand product 2 service bad 3 customer unhappy fact satisfied think time give false impression 4 support team 5 link product sentence sentiment rating output satisfactory sentence positive number 3 result reliable run example time let code level find click view code playground copy paste sentimentanalysis ipynb chapter notebook

add line print want response = openai.completion.create prompt="this sentence sentiment classifier\nsentence \"the customer satisfied\"\nsentiment positive\n###\nsentence \"the customer satisfied\"\nsentiment negative\n###\nsentence \"the \"\nsentiment positive\n###\nsentence \"this link review\"\nsentiment neutral\n###\nsentence text\n\n\n1 \"i love new batman movie!\"\n2 \"i hate phone battery dies\"\ n3 \"my day \"\n4 \"this link article\"\n5 \"this new music video blow mind\"\n\n\nsentence sentiment ratings:\ n1 positive\n2 negative\n3 positive\n4 neutral\n5 positive\n\n\n###\ nsentence

text\n\n\n1 \"i stand product\"\n2 \"the service \"\n3 \"though customer unhappy fact satisfied think time give false impression\"\n4 \"the support team \"\n5 \"here link product.\"\n\n\nsentence sentiment ratings:\n detect customer emotion prediction r = response["choices"][0 output stable follow response run 1 sentence number 3 neutral run 2 sentence number 3 positive run 3 sentence number 3 positive run 4 sentence number 3 negative lead conclusion investigation srl show sentence simple complete ellipsi miss word reliable sentiment analysis output srl show sentence moderately difficult output srl show sentence complex ellipsi proposition ambiguous phrase solve result stable reliable conclusion

job position developer present future ai development require cloud ai ready use module design skill require classical development pipeline feed ai algorithm control analyze output require thinking target development chapter show huge future developer thinker designer pipeline development time sum journey explore new transformer horizon chapter go advanced theory principle compositionality intuitive concept principle compositionality mean transformer model understand sentence understand sentence involve logical form rule provide link sentence segment theoretical difficulty sentiment analysis require large transformer model training powerful machine human resource transformer model train task require training specific task test roberta large distilbert minilm l12 h384 uncased excellent bert base multilingual model find provide interesting answer require training solve sst sample run model sentiment analysis require deep understanding sentence extraordinarily complex sequence sense try roberta large mnli interference task produce lesson conventional unconventional trans- model try try different model task transformer flexibility allow try different task model task different model gather idea way improve customer relation detect customer unsatisfied customer seek competition custom- er complain product service anticipate future problem improve service display quality service online real time representation detect customer emotion prediction finally run sentiment analysis gpt-3 directly online use interface surprisingly effective human require solve difficult sequence see srl help identify

effective human require solve difficult sequence see srl help identify issue complex sequence conclude developer huge future thinker designer pipeline devel- chapter analyze fake news transformer use sentiment analysis analyze emotional reaction fake news necessary pretrain transformer sentiment analysis true false 2 sentence positive negative neutral true false principle compositionality signify transformer grasp sentence understand true false roberta large design improve pretraine process transformer model 5 transformer provide feedback inform customer satisfied sentiment analysis product service consistently negative help

proper decision improve offer true false model fail provide good result task require training change richard socher

alex perelygin jean wu jason chuang christopher manning andrew ng christopher potts recursive deep models semantic compositionality sentiment hugging face pipeline model documentation yinhan liu danqi chen omer levy mike lewis luke zettlemoyer veselin stoyanov 2019 roberta robustly optimize bert pretraining approach https://arxiv.org/ allen institute ai https://allennlp.org/ allen institute reading comprehension resource https://demo.allennlp.org/ roberta large contribution zhaofeng wu https://zhaofengwu.github.io/ stanford sentiment treebank https://nlp.stanford.edu/sentiment/treebank.html

join book discord space join book discord workspace monthly ask session author analyze fake news bear think earth flat baby crawl flat surface kin- dergarten child play flat playground elementary school sit flat classroom parent teacher tell earth round people upside take understand fall earth today beautiful sunset sun set earth rotate away sun take time effort figure fake news like child work way perceive fake news chapter tackle topic create tension check fact topic climate change gun control donald trump tweet analyze tweet face- book post source information goal certainly judge anybody fake news involve opinion fact news depend perception fact local culture provide idea tool help gather information topic find way jungle infor- mation receive daily focus ethical method performance transformer gpt-3 engine reason replace human judgment instead provide tool human judgment manually gpt-3 engine attain human level performance task leave moral ethical decision making human analyze fake news transformer begin define path lead react emotionally ratio- nally fake news define method identify fake news transformer heuristic resource build previous chapter understand explain fake news judge provide transformer model explain news prefer create universal absolute transformer model detect assert message choose educate user transformer lecture

approach opinion fact chapter cover following topic emotional reaction fake news behavioral representation fake news rational approach fake news fake news resolution roadmap apply sentiment analysis transformer task social medium analyze gun control perception ner srl information extract transformer find reliable website transformer produce result educational purpose read president trump tweet objective critical eye step explore emotional rational reaction fake news emotional reaction fake news human behavior tremendous influence social cultural economic decision emotion influence economy rational thinking behavioral economic drive decision making process buy consumer good physically need satisfy emotional desire buy smartphone heat moment exceed budget emotional rational reaction fake news

smartphone heat moment exceed budget emotional rational reaction fake news depend think slowly react quickly incoming information daniel kahneman describe process research book think fast slow 2013 vernon l. smith award nobel memorial prize economic sciences behavioral economic research behavior drive decision previously think rational unfortunately decision base emotion reason let translate concept behavioral flowchart apply fake news cognitive dissonance trigger emotional reaction cognitive dissonance drive fake news rank twitter facebook social medium platform everybody agree content tweet happen some- body write tweet say climate change important react enter state cognitive dissonance tension build contradictory thought mind result nervous agitated wear like short circuit toaster example think wear mask covid-19 necessary outdoors lockdown good bad thing coronavirus vaccine effective coronavirus vaccine dangerous cognitive dissonance like musician keep make mistake play simple song drive crazy fake news syndrome increase cognitive dissonance exponentially expert assert vaccine safe need careful expert say wear mask outside useless assert news channel wear accuse fake news appear significant portion fake news truth 2022 republicans democrats unable agree national election rule wake 2020 presidential election organization upcoming election find score topic open newspaper read an- view oppose common sense premise chapter draw example find transformer model automatically detect

fake news make sense world social medium multicultural expression group sense know truth group express fake news try express view truth culture make sense global world culture vary country continent social analyze fake news transformer fake news absolute myth need find well definition fake news opinion fact course fake news state cognitive dissonance resolve cognitive reasoning resolve problem fake news exactly like try resolve conflict party mind chapter life recommendation analyze conflictual tension deconstruct- e conflict idea transformer model combat fake news find inner peace pretend use transformer find absolute truth oppose fake news use transformer obtain deep understanding

absolute truth oppose fake news use transformer obtain deep understanding sequence word message form profound broad opinion topic let lucky user transformer model obtain well vision opinion design chapter classroom exercise use transformer great way deepen understanding language sequence form broad- er opinion develop cognitive ability let start see happen somebody post conflictual tweet analyze conflictual tweet follow tweet message post twitter paraphrase tweet show chapter raw dataset format twitter interface display sure people disagree content lead political figure famous actor tweet climate change bogus plot liberal economy trigger emotional reaction tweet pile side viral trend let run tweet transformer tool understand tweet create cognitive dissonance storm somebody mind open fake_news.ipynb notebook section begin resource allen institute ai run roberta transformer model sentiment analysis chapter 12 detect customer emotion prediction install allennlp model pip install allennlp==1.0.0 allennlp models==1.0.0 allennlp continuously update version progress version 2.4.0 exist time writing provide additional value example provide chapter date stochastic algorithm model update produce different output run cell bash analyze output tweet detail information model output echo sentence":"climate change bogus plot liberal economy | allennlp predict https://storage.googleapis.com/allennlp-public-models/ output show tweet negative positive value 0 negative value prob 0.0008486526785418391 0.999151349067688

output vary run transformer stochastic algorithm https://allennlp.org/ visual representation analysis output change run transformer model continuously train update goal chapter focus reasoning transformer model select sentiment analysis https://demo.allennlp.org/sentiment-analysis choose roberta large model run analysis obtain negative result investigate word influence roberta decision model interpretation interpret model provide insight result obtain choose look option simple gradient visualization approach provide visualization compute gradient score class relate input second saliency main feature map infer class input integrated gradient visualization model require change neural network motivation design call gradient generate attribution prediction neural network input analyze fake news transformers smooth gradient visualization approach compute gradient output prediction input goal identify feature input noise add improve interpretation section model interpretations click simple gradient visualization interpret prediction obtain following representation figure 13.1 visualize 3 important word + bogus + plot influence negative prediction point wonder look simple example explain cognitive dissonance explanation come follow tweet staunch republican write tweet let member jaybird65 surprise fellow republican tweet follow tweet republican think climate change consciousness great thing tweet come member hunt78 let run sentence fake_news.ipynb echo sentence":"i republican think climate change consciousness great thing | allennlp predict https://storage.googleapis.com/allennlp-public-models/ output positive course prob 0.9994876384735107 0.0005123814917169511 cognitive dissonance storm build jaybird65 mind like hunt78 disagree mind storm intensify read subsequent tweet ensue jaybird65 hunt78 discover surprising fact hurt jaybird65 feeling jaybird65 hunt78 obviously know respective twitter account hunter staunch republicans jaybird65 initial tweet come reaction article new york times state climate change destroy planet jaybird65 puzzled hunt78 republican like hunter hunt78

believe climate change twitter thread go massive number rage tweet root fake news discussion lie emotional reaction news rational approach climate change simply matter cause climate change need economy change human need continue build electric car walking space large city well agricultural habit need business new way probably emotion strong human let represent process lead news emotional rational reaction behavioral representation fake news fake news start emotional reaction build lead personal attack figure 13.2 represent phase emotional reaction path fake news cognitive dissonance clog thinking process phase 1 income news person group person react news obtain respective medium facebook twitter social medium tv radio website source information contain phase 2 consensus person group person agree disagree disagree enter phase 3 conflict rage analyze fake news transformer agree consensus stop heat build news accept real news party believe news receive fake mean fake thing explain news label fake news early 12 th century people europe agree earth center universe solar system rotate earth 1900 people believe thing airplane fly ocean january 2020 europeans believe covid-19 virus impact china global pandemic line consensus party society mean incoming news true false party disagree lead conflict figure 13.2 representation path news fake news conflict let face social medium member usually converge idea rarely change mind matter representation show person stick opinion express tweet conflict escalate soon somebody challenge message phase 3 conflict fake news conflict divide phase 3.1 conflict begin disagreement party tweet post message facebook platform exchange conflict wear party interested topic 3.2 climate change discussion jaybird65 hunt78

know thing nasty conversation heat 3.3 point inevitably argument party fake news jaybird65 angry numerous tweet climate change hu- mans fake news hunt78 angry deny human contribution climate change fake news 3.4 discussion end personal attack godwin law enter con- versation know get godwin law state party find bad reference possible describe party point conversation come liberal like hitler try force economy climate change type message see twitter facebook platform appear real time chat presidential speech climate rational approach

discussion soothe party calm reach middle ground consensus forward let try build rational approach transformer heuristic rational approach fake news transformer powerful nlp tool section define method party engage conflict fake news emotional level rational level use transformer tool heuristic run transformer sample gun con- trol president trump tweet covid-19 pandemic describe heuristic implement classical function analyze fake news transformer implement transformer nlp task task choice case roadmap method help teacher parent friend co worker anybody seek truth work worthwhile let begin roadmap rational approach fake news include transformer define fake news resolution roadmap figure 13.3 define roadmap rational fake news analysis process process contain transformer nlp task traditional function figure 13.3 go emotional reaction fake news rational representation rational process nearly begin emotional reaction begin rational process kick soon possible avoid build emotional reaction interrupt discussion phase 3 contain tool 3.1 sentiment analysis analyze rank emotional positive negative word use allennlp resource run roberta large transformer fake_news ipynb notebook use allennlp visual tool visualize keyword ex- planation introduce sentiment analysis chapter 12 detect customer emotion prediction 3.2 name entity recognition ner extract entity social medium message phase 3.4 describe ner chapter 11 let data talking story question answer use hugging face bert transformer model task addition use allennlp.org visual tool visualize entity explanation 3.3

semantic role labeling srl label verb social medium message phase 3.4 describe srl chapter 10 semantic role labeling bert base transformer use allennlp bert model fake_news.ipynb use allennlp.org visual tool visualize output labeling task 3.4 reference reliable website describe classical coding help let begin gun control gun control debate second amendment constitution united states assert following right regulate militia necessary security free state right people bear arm shall infringe america divide subject decade hand argue right bear firearm want endure gun control argue fake news contend possess weapon hand argue bear firearm dangerous gun control remain violent country argue fake news contend dangerous

carry weapon need help party let begin sentiment analysis analyze fake news transformer read tweet facebook message youtube chat speech social medium party fight rage battle need tv eat popcorn tweet battle tear party apart let tweet facebook message opposing change member name paraphrase text bad idea consider insult message let start pro gun tweet tweet honest opinion person afirst78 rifle gun year problem raise kid right gun hurt rabbit let run fake_news.ipynb echo sentence rifle gun year problem raise kid right gun hurt rabbit | allennlp predict https://storage.googleapis.com/allennlp-public-models/ prediction positive prediction logit 1.9383275508880615 -1.6191326379776 prob visualize result allennlp simple gradient visualization provide ex- figure 13.4 simple gradient visualization sentence explanation show sentiment analysis tweet afirst78 highlight rifle + + rabbit pick idea function step fake_news_function_1 function fake_news_function_1 rifle + + rabbit extract note analysis rifle dangerous example analyze nys99 view gun control gun control analysis nys99 hear gunshot life neighborhood lose friend afraid night let run analysis fake_news.ipynb echo sentence hear gunshot life neighborhood lose friend afraid night | allennlp predict https://storage.googleapis.com/allennlp-public-models/ result naturally negative prediction logit -1.3564586639404297 0.5901418924331665 prob 0.12492450326681137 0.8750754594802856

let find keyword allennlp online run sample smooth gradient visualization highlight follow figure 13.5 smooth gradient visualization sentence keyword afraid stand function 2 section know afraid as- sociate gun result vary run time transformer model continuously train update focus chapter pro- cess specific result analyze fake news transformer model problem interpret cognitive dissonance human critical thinking necessary fake_news_function_2 afraid gun topic extract note anal- function clearly understand party fight fake_news_function_1 rifle + + rabbit

afirst78 probably live mid western state state small population quiet enjoy low crime rate afirst78 travel major city enjoy pleasure quiet life country fake_news_function_2 afraid + topic

gun nys99 probably live big city great area major city crime rate high violence daily phenomenon nys99 travel mid western state see afirst78 live honest strong view prove need implement solution like de- scribe chapter well information key few fake news battle follow process apply name entity recognition sentence name entity recognition ner chapter show transformer method user benefit broad perception message different angle html page sum chapter transformer method contain transformer task production mode apply process tweet facebook message entity message program know run message illustrate step process install hugging face transformer pip install -q transformer transformer import pipeline transformer import autotokenizer run message nlp_token_class = pipeline('ner nlp_token_class('i rifle gun year problem raise kid right gun hurt rabbit output produce result entity mean take pipeline sentence contain location person provide clue culture area let check model output show model use 9 label 1,024 feature attention layer analyze fake news transformer bert 24 layer transformer model wish explore architecture run run srl message semantic role labeling srl continue run fake_news.ipynb cell cell order find notebook examine point view let start pro gun perspective run following cell fake_news.ipynb echo sentence rifle gun year problem raise kid right gun hurt rabbit | allennlp predict https://storage.googleapis.com/allennlp-public-models/ output detailed useful wish investigate parse label detail show excerpt prediction verbs verb description arg0 v arg1 rifle gun argm tmp year problem let visual detail allennlp semantic role labeling section run srl task message verb show afirst78 experienced gun owner figure 13.6 srl verb argument sum afirst78 experience + rifle gun + year second frame add information + + + problem argument raise display afirst78 parental experience figure 13.7

+ problem argument raise display afirst78 parental experience figure 13.7 srl verb argument verb raise analyze fake news transformer argument explain pro gun position kid + gun hurt result vary run model update process remain add find collection function parsing fake_news_function_3 + rifle gun + year fake_news_function_4 kid + gun hurt let explore gun

control message gun control srl run facebook message fake_news.ipynb continue run notebook cell cell order create notebook echo sentence hear gunshot life neighborhood lose friend afraid night |

allennlp predict https://storage.googleapis.com/allennlp-public-models/ result label key verb sequence detail show follow excerpt prediction verb verb hear description arg0 v hear arg1 gunshot life neighborhood continue apply process allennlp semantic role labeling section enter sentence run transformer model verb hear show tough reality message figure 13.8 srl representation verb hear quickly parse word fifth function fake_news_function_5 hear + gunshot + life verb lose show significant argument relate figure 13.9 srl representation verb lose need sixth function fake_news_function_6 lose + + friend good suggest reference site user different transformer model clarify aspect message run transformer nlp task describe traditional heuristic hard coding need develop parse datum generate function pro gun fake_news_function_1 + problem + gun gun control fake_news_function_2 hear + afraid + gun pro guns fake_news_function_3 + rifle gun + year pro guns fake_news_function_4 kid + gun + hurt bear mind result vary run function generate different time provide slightly different result previous section main idea remain let focus function analyze fake news transformer gun control fake_news_function_5 hear + gunshot + life gun control fake_news_function_6 lose + + friend let reorganize list separate perspective draw conclusion decide pro gun gun control pro gun argument honest lack information go major city pro gun fake_news_function_1 + problem + gun pro gun fake_news_function_3 + rifle gun + year pro gun fake_news_function_4 kid + gun + hurt gun control argument honest lack information large quiet area midwest gun control fake_news_function_2 hear + afraid + gun gun control fake_news_function_5 hear + gunshot + life gun control fake_news_function_6 lose + + friend function develop inform party example let function1 express pseudocode function 2 + 5 + 6 keyword simplify google search = afraid gun lose friend gunshots goal process run transformer model deconstruct explain message nlp transformer like mathematical calculator produce good result take free think

human mind

mathematical calculator produce good result take free think human mind interpret ask train nlp human user proactive search read information well transformer model help user understand message deeply think try help user lecture brainwash parse require process result function hundred social medium message automatically let program job link change google modify search link appear interesting pro gun advocate figure 13.10 gun violence analyze fake news transformer let imagine search gun control advocate following pseudocode function 1 + 3 + 4 keyword simplify google search = problem gun year kid hurt google search return clear positive result favor pro gun advocate interesting one neutral educational figure 13.11

gun safety run automatic search amazon bookstore magazine educational importantly essential people oppose idea talk get fight understand good way develop empathy side tempt trust social medium company recommend let party act proxy mind process use transformer model deconstruct message remain proactive consensus topic agree follow safety guideline gun possession example choose gun home lock safely child access let covid-19 president trump tweets covid-19 president trump tweet matter political opinion say donald trump donald trump book analyze information technical political book analyze tweet scientifically describe educational approach fake news gun control section chapter need process implement run allennlp srl task bert model fake_news.ipynb note- book gun control section section focus logic fake news run bert model srl visualize result allennlp website let presidential tweet covid-19 semantic role labeling srl srl excellent educational tool tend read tweet passively listen break message srl good way develop social medium analytical skill distinguish fake accurate information recommend srl transformer educational purpose class young student en- ter tweet analyze verb argument help young generation active reader social media analyze relatively undivided tweet conflictual tweet let analyze late tweet find july 4 write book take person refer black american paraphrase president text x great american

hospitalize coronavirus request prayer join pray today suffer analyze fake news transformer let allennlp semantic role labeling section run sentence look result verb hospitalize show member stay close fact figure 13.12 srl argument verb hospitalize message simple x + hospitalize + coronavirus verb request show message political figure 13.13 srl argument verb request know person request president pray decide center request good exercise display html page ask user think exam- ple user ask look result srl task answer following president trump ask pray deviate request political reason fact president trump state indirectly ask pray x fake news think decide let look

indirectly ask pray x fake news think decide let look ban twitter take name paraphrase tone run allennlp visualize result surprising srl tone paraphrase tweet thug dishonor memory x. looting start action take suppress main original tweet srl task show bad association tweet figure 13.14 srl argument verb dishonoring educational approach explain associate argument thug memory looting fit important exercise ask user srl argument fit recommend exercise transformer model user develop srl skill critical view topic present critical thinking good way stop propagation fake news pandemic go rational approach fake news transformer heuristic in- structive website end lot heat fake news debate boil emotional irrational reaction analyze fake news transformer world opinion find entirely objective transformer model detect fake news oppose side agree truth place agree transformer model output model bias build enemy opinion good approach listen keep heat chapter focus apply transformer problem find silver bullet transformer model exist main option solve nlp problem find new transformer model create reliable durable method implement transformer model conclude chapter interpret transformer model fake news begin deep inside emotional history human event occur emotion help react quickly situation hardwire react strongly fake news spur strong reaction fear news temporarily permanently damage life believe climate change eradicate human life earth believe react strongly climate change destroy economy break society believe gun dangerous remind second amendment united states constitution give right possess gun go

rage conflict covid-19 president trump climate change case see emotional reaction fast one build conflict design roadmap emotional perception fake news rational level transformer nlp task possible find key information tweet facebook message medium new perceive real news fake news create rationale teacher parent friend co worker people talk add classical software func- tion help way point toolkit transformer model nlp task sample dataset use artificial intelligence good humanity transformer

nlp task sample dataset use artificial intelligence good humanity transformer tool idea implement world well place good way understand transformer visualize internal process analyze transformer gradually build representation sequence chapter interpret black box transformer models news label fake news fake true false news everybody agree accurate true false transformer run sentiment analysis tweet true false key entity extract facebook message distilbert model run key verb identify youtube chat bert base model run srl emotional reaction natural response fake news true false rational approach fake news help clarify position true false 8 connect transformer reliable website help somebody understand news fake true false transformer summary reliable website help understand topic label fake news true false 10 change world use ai good true false daniel kahneman 2013 think fast slow hug face pipeline https://huggingface.co/transformers/main_classes/ allen institute ai https://allennlp.org/ analyze fake news transformer join book discord space join book discord workspace monthly ask session author interpret black box million- billion parameter transformer model like huge black box interpret result developer user discourage deal mind blow model recent research begin solve problem innovative cutting edge tool scope book describe explainable ai method algorithm instead chapter focus ready use visual interface provide insight transformer model developer user chapter begin instal run bertviz jesse vig jesse excellent job build visual interface show activity attention head bert transformer model bertviz interact bert model provide design interactive interface continue focus visualize activity transformer model language interpretability tool lit lit non probing tool use pca umap represent transformer

model prediction pca use umap finally visualize transformer journey layer bert model dictionary learning local interpretable model agnostic explanations lime provide practical function visualize transformer learn understand language method show transformer begin learn word word sentence context finally long range dependency interpret black box transformer model end chapter able interact

dependency interpret black box transformer model end chapter able interact user visualization activity transformer model bertviz lit visualization dictionary learning long way nascent tool help developer user understand transformer model work chapter cover following topic instal run bertviz run bertviz interactive interface difference probe non probing method principal component analysis pca reminder run lit analyze transformer output run transformer visualization dictionary learning word level polysemy disambiguation visualize low level mid level high level dependency visualize key transformer factor step begin instal bertviz transformer visualization bertviz jesse vig article multiscale visualization attention transformer model 2019 recognize effectiveness transformer model jesse vig explain decipher atten- tion mechanism challenging paper describe process bertviz visualization tool bertviz visualize attention head activity interpret transformer model behavior bertviz design visualize bert gpt-3 model section visualize activity bert model let install run bertviz take step visualize transformer attention head interact open bertviz.ipynb notebook chapter14 directory github repository book step install bertviz requirement step 1 instal bertviz import module notebook install bertviz hugging face transformer basic requirement implement program pip install bertviz bertviz import head_view model_view transformer import berttokenizer bertmodel head view model view library import load bert model step 2 load model retrieve attention bertviz support bert gpt-2 roberta model consult bertviz github information https://github.com/jessevig/bertviz section run bert base uncase model pretrained tokenizer load model retrieve attention model_version = bert base uncased do_lower_case = true model = bertmodel.from_pretrained(model_version

output_attention = true tokenizer = berttokenizer.from_pretrained(model_version do_lower_case = enter sentence try different sequence analyze behavior model sentence_b_start necessary step 5 model view sentence_a = lot people like animal adopt cat sentence_b = lot people like animal adopt dog input = tokenizer.encode_plus(sentence_a sentence_b return token_type_id = inputs['token_type_id input_id = inputs['input_ids attention = model(input_id token_type_id = token_type_ids)[-1 sentence_b_start = token_type_ids[0].tolist().index(1 input_id_list = input_ids[0].tolist batch index 0

token = tokenizer.convert_ids_to_tokens(input_id_list ready interact visualization interface interpret black box transformer model step 3 head view final line add activate visualization attention head word layer layer 0 actual token interface educational 12 attention head layer display different color default view set layer 0 show figure 14.1 figure 14.1 visualization attention head ready explore attention head step 4 process display attention head color column token represent attention head layer number choose layer number click attention head color word sentence break token attention section word token loosely refer word help understand transformer head work focus word animal figure 14.2 figure 14.2 select layer attention head token bertviz show model connection animal word normal layer 0 layer 1 begin isolate word animal relate show figure 14.3 figure 14.3 visualize activity attention head 11 layer 1 interpret black box transformer models attention head 11 make connection animal people adopt click cat interesting connection show figure 14.4 figure 14.4 visualize connection cat token word cat associate animal connection show model learn cat animal change sentence click layer attention head visualize transformer make connection find limit course good bad connec- tion transformer work fail case valuable explain transformer behave require layer parameter datum let bertviz display model view step 5 model view take line obtain model view transformer bertviz model_view(attention token sentence_b_start bertviz display layer head view show view excerpt figure 14.5 figure 14.5 model view mode bertviz click head obtain

head view word word sen- tence sentence option attention head transformer model make well representation progress layer example figure 14.6 show activity attention head layer figure 14.6 activity attention head low layer model representation make connection separator sep word sense token activate atten- tion head layer level

training transformer model limit quality case bertviz remain interesting educational tool interpretability tool trans- let run intuitive lit tool interpret black box transformer models lit visual interface help find example model process incorrectly dig similar example model behave change context language issue relate transformer model lit display activity attention head like bertviz worth analyze thing go wrong try find solution choose uniform manifold approximation projection umap visualization pca projector representation pca linear projection specific direction magnitude umap break projection mini cluster approach sense depend far want analyze output model run obtain different perspective model example section use pca run lit let begin brief reminder pca work pca take datum represent high level imagine kitchen kitchen 3d cartesian coordinate system object kitchen specific x y z coordinate want cook recipe gather ingredient kitchen table kitchen table high level representation recipe kitchen kitchen table cartesian coordinate system extract main fea- ture

kitchen represent recipe kitchen table perform pca display principal component fit specific recipe representation apply nlp example dictionary list word word mean constitute representation principal component sequence pca representation sequence lit help visualize output transformer main step obtain nlp pca representation variance numerical variance word dataset frequency frequency meaning example covariance variance word relate word eigenvalue eigenvector obtain representation cartesian system need vector magnitude representation covariance eigenvector provide direction vector eigenvalue provide magnitude derive datum step apply feature vector original dataset multiply row feature vector row datum datum display = row feature vector row datum pca projection provide clear linear visualization

data point analyze let run lit run lit online open google colaboratory notebook click following link access option tutorial page contain type nlp task analyze section run lit online explore sentiment analysis classifier click explore demo enter intuitive lit interface transformer model small transformer model figure 14.7 select model change model click model test type model similar one directly hugging face host api page nlp model change lit online version base subsequent update concept remain model change interpret black box transformer model let begin select pca projector binary 0 1 classification label sentiment analysis example figure 14.8 select projector type label data table click sentence classification label figure 14.9 select sentence algorithm stochastic output vary run sentence appear datapoint editor figure 14.10 datapoint editor datapoint editor allow change context sentence example want find go wrong counterfactual classification class end change context sentence appear correct class understand model work mistake sentence appear pca projector classification figure 14.11 pca projector positive cluster click data point pca projector sentence appear datapoint editor sentence select way compare result lit contain wide range interactive function explore use let visualize transformer layer dictionary learning transformer visualization dictionary learning transformer visualization dictionary learning base transformer factor

visualization dictionary learning transformer visualization dictionary learning base transformer factor transformer factor embed vector contain contextualize word word context meaning create polysemy issue example word separate verb adjective furthermore separate mean disconnect discriminate scatter definition result obtain lit convincing lit provide valu- able insight case essential involve emerge tool technique interpret black box transformer models yun et al 2021 create embed vector contextualize word word embed- ding vector construct sparse linear representation word factor example depend context sentence dataset separate represent separate=0.3 apart"+"0.3 distinct"+ 0.1 discriminate"+0.1 sever + 0.1 disperse"+0.1 scatter ensure linear representation remain sparse add 0 factor create huge matrix 0 value include useless information separate= 0.0"putting together"+".0 identical point

representation sparse force coefficient factor great 0 hidden state word retrieve layer layer progress under- standing representation word dataset sentence latent dependency build sparse linear superposition transformer factor dictionary matrix sparse vector coefficient infer sum phi dictionary matrix sparse vector coefficient infer yun et al 2021 add gaussian noise sample force algorithm search deep ensure representation remain sparse equation write s.t $\alpha > 0$ author refer x set hide state layer x sparse linear superposition transformer factor belong x. beautifully sum sparse dictionary learning model dictionary matrix :,c refer column dictionary matrix contain trans- :,c divide level low level transformer factor solve polysemy problem word level mid level transformer factor sentence level pattern bring vital context low level high level transformer pattern help understand long range dependency method innovative exciting efficient visualization func- tionality point yun et al 2021 create necessary information lime standard interpretable ai method visualize finding interactive transformer visualization page base lime output follow section brief introduction lime lime stand local interpretable model agnostic explanations explain- able ai method speak model agnostic draw immediate consequence method transformer visualization dictionary learning method dig

draw immediate consequence method transformer visualization dictionary learning method dig matrix weight matrix multiplication trans- method explain transformer model work chapter 2 get start architecture transformer model chapter method peek mathematical output provide sparse linear superposition transformer factor lime try parse information dataset instead lime find model locally reliable examine feature prediction lime apply model globally focus local environment prediction particularly efficient deal nlp lime explore context word provide invaluable information model output visualization dictionary learning instance x represent interpretable representation instance binary vector interpret black box transformer model goal determine local presence absence feature feature nlp feature token reconstruct word lime g represent transformer model machine learning model g represent set

transformer model contain g model lime algorithm apply transformer model point know lime target word search local context word lime provide local context word explain word predict explore explainable ai lime scope book transformer nlp lime reference section let lime fit method transformer visualization dictionary learning let explore visualization interface visualization interface visit following site access interactive transformer visualization page https:// visualization interface provide intuitive instruction start analyze transformer factor specific layer click show figure 14.12 figure 14.12 select transformer factor choose factor click layer wish visualize factor figure 14.13 visualize function layer visualization show activation factor layer layer figure 14.14 importance factor layer interpret black box transformer models factor 421 focus lexical field separate show low layer figure 14.15 representation separate low layer visualize high layer long range representation emerge factor 421 begin representation separate high level transformer begin form deep under- standing factor associate separate distinct

show figure 14.16 figure 14.16 high layer representation transformer factor try transformer factor visualize transformer expand perception un- derstanding language layer layer find good example poor result focus good example understand transformer make way language learning use poor result understand mistake transformer model visualization interface powerful train case involved stay loop evolve field example explore understanding_gpt_2_models_with_ecco.ipynb github repository book chapter show transformer generate candi- date choose token self explanatory section see transformer learn meaning word layer layer transformer generate candidate make choice show notebook transformer model stochastic choose probability consider following sentence sun rise word choose end sentence hesitate transformer case gpt-2 model choose word sky figure 14.17 complete sequence candidate gpt-2 model choose run show figure figure 14.18 candidate completion sky appear rank morning appear rank second fit run model time obtain different output model stochastic domain ai transformer complete let human lot work interpret black

box transformer model explore model access visual interface explore chapter fascinating lot work example openai gpt-3 model run online api access weight software service saas transformer model trend increase expand year come corporation spend million dollar research computer power tend provide pay service open source application access source code output weight gpt-3 model visual in- terface analyze 9,216 attention head 96 layer x 96 head challenging find wrong require human involvement case example polysemy issue word coach english french translation rep- resent problem english coach person train people bus word coach exist french apply person train people openai ai gpt-3 playground https://openai.com/ translate sentence con- taine word coach obtain mixed result sentence 1 translate correctly openai engine english coach break everybody complain french le bus eu un problème et tout le monde s'est plaint coach translate bus fine context require output stochastic translation correct time false time sentence 2

mistranslate english coach dissatisfied team everybody french le bus était insatisfait du équipe et tout le monde s'est plaint time gpt-3 engine miss fact coach mean person bus stochastic run provide unstable output modify sentence 2 add context obtain proper translation english coach football team dissatisfied everybody

french le coach de l'équipe de football était insatisfait et tout le monde s'est plaint translation contain french word coach definition english word coach sentence context add openai solution ai general transformer model particular continuously pro- gressing furthermore industry 4.0 ai drive micro decision require level sophistical nlp translation task effective human intervention development cloud ai api level remain nec- essary long time transformer model train resolve word level polysemy disambiguation low level mid level high level dependency process achieve connect train million- trillion parameter model task interpret giant model daunting tool emerge instal bertviz learn interpret computation attention head interactive interface see word interact word layer chapter continue define scope probe non

probing task probe task ner provide insight transformer model represent language non probing method analyze model make prediction example lit plug pca project umap representation output bert transformer model analyze cluster output fit finally run transformer visualization dictionary learning user choose transformer factor analyze visualize evolution representation low layer high layer transformer factor progressively polysemy disambiguation sentence context analysis finally long term dependency tool chapter evolve technique key takeaway chapter transformer model activity visualize interpret user friendly manner chapter discover new transformer model risk management method choose good implementation transformer model project interpret black box transformer models bertviz show output layer bert model true false bertviz show attention head layer bert model true false bertviz show token relate true false lit show inner working attention head like bertviz true false probing way algorithm predict language representation true false ner probe task true false pca umap non probing task true false 8 lime model agnostic true false transformer deepen relationship token layer layer true false 10 visual transformer model interpretation add new dimension interpretable ai true/

bertviz jesse vig,2019 multiscale visualization attention transformer model,2019 lit explanation sentiment analysis representation https://pair-code.github.io/ transformer visualization dictionary learning zeyu yun yubei chen bruno olshau- sen yann lecun 2021 transformer visualization dictionary learning contextual- ize embedding linear superposition transformer factor https://arxiv.org/

transformer visualization dictionary learning https://transformervis.github join book discord space join book discord workspace monthly ask session author nlp task agnostic examine variation original transformer model encoder decoder layer explore model encoder decoder stack layer size layer parameter increase fundamental architecture transformer retain original structure identical layer parallelization computing attention head chapter explore innovative

transformer model respect basic structure original transformer significant change score transformer model appear like possibility box lego hundred way transformer model sublayer layer lego begin ask transformer model choose offer ecosystem implement discover locality sensitivity hashing lsh bucket chunking reformer model learn disentanglement deberta model deberta introduce alternative way manage position decoder deberta high powered transformer model exceed human baseline step stop discover powerful computer vision transformer vit clip dall e. add clip dall e openai gpt-3 google bert train google small group foundation model nlp task agnostic transformer models powerful foundation model prove transformer task agnostic transformer learn sequence sequence include vision sound type datum represent sequence image contain sequence data like language run vit clip dall e model learn vision model innovative level end chapter world task agnostic transformer evolve universe imagination creativity chapter cover following topic choose transformer model reformer transformer model locality sensitivity hashing lsh bucket chunking technique deberta transformer model text image vision transformer clip dall e creative text image vision transformer step choose model ecosystem choose model ecosystem think test transformer model download require machine human resource think platform online sandbox time risk work test example site hugging face download pretraine model automatically real time reformer deberta section thank run hugging face model google colab instal machine test hugging face model online idea analyze have install install 2022 mean run transformer task online run transformer preinstalle google colaboratory vm seamlessly down- load pretrained model task run line run transformer api definition

load pretrained model task run line run transformer api definition install expand past year definition online widen consider line code run api meta online test refer instal online broad sense section figure 15.1 show test model online figure 15.1 test transformer model online test decade flexible productive follow show hugging face host api model deberta model addition hugging face offer automl service train deploy transformer model openai gpt-3

engine run online playground provide api openai offer model cover nlp task model require training gpt-3 billion pa- rameter zero shot engine impressive show transformer model parameter produce well result overall microsoft azure google cloud ai allennlp platform offer interesting service online model analysis read paper worthwhile good ex- ample google publication fedus et al 2021 switch transformers scale trillion parameter model simple efficient sparsity google increase size t5 base model study chapter 8 apply transformer legal finan- cial document ai text summarization paper confirm strategy large online model gtp-3 end take risk choose solution time spend explore platform model help optimize implementation project choice host choice different way show figure 15.2 local machine api openai google cloud ai microsoft azure ai hugging face provide good api application local machine cloud platform cloud service api nlp task agnostic transformer models cloud platform amazon web services aws google cloud train fine tune test run model platform case application local machine cloud api local machine data center vm mean api integrate physical system windmill airplane rocket autonomous vehicle system permanently connect system api figure 15.2 implement option model end decision time test analyze compute cost work team listen different perspective understand transformer work well choice let explore reformer variation original transformer model kitaev et al 2020 design reformer solve attention memory issue add functionality original transformer model reformer solve attention issue locality sensitivity hashing

original transformer model reformer solve attention issue locality sensitivity hashing lsh bucket lsh search near neighbor dataset hash function determine datapoint q close p hash(q = = hash(p case data point key transformer lsh function convert key lsh bucket b1 b4 figure 15.3 process call lsh bucketing like object similar sort bucket sort bucket split chunk c1 c4 figure 15.3 parallelize finally attention apply bucket chunk previous chunk figure 15.3 lsh attention head lsh bucketing chunking considerably reduce complexity o(l 2 attend word pair o(llogl attend content bucket reformer solve memory issue recompute layer input instead store

information multi layer model recomputing achieve demand instead consume terabyte memory

large multi layer model use reformer model train english translation crime punishment fyodor dostoevsky run example let run directly online host inference api input sentence student impoverished know link online interface contain input nlp task agnostic transformer models host inference api appear input sentence click compute obtain inference result appear right input figure 15.4 reformer host inference api different response algorithm stochastic reformer reasonably train supercomputer billion bit information like openai gpt- 3 result reformer impressive training fine tuning obtain well result openai gpt-3 engine produce following result text completion student impoverished know turn find place stay take pad bag start write write xxxxxxxxxx student xxxxxxxxxx family friend result convincing access openai playground having sign show highly train transformer model contain billion parameter outperform innovative transformer model architecture supercomputer drive cloud ai platform progressively outperform local attempt powerful cloud platform need address issue prototype in- vest solution note openai gpt-3 transformer model deep learning model base stochastic algorithm result vary deberta introduce innovative architecture explore new approach transformer find disentanglement disentanglement ai allow separate representation feature training process flexible pengcheng xiaodong liu

jianfeng gao weizhu chen design deberta disentangled version transformer describe model interesting article deberta decode en- hance bert disentangle attention https://arxiv.org/abs/2006.03654 main idea implement deberta disentangle content position transformer model train vector use absolute position decoder predict mask token pretraining process author provide code github https://github.com/microsoft/deberta deberta exceed human baseline superglue leaderboard figure 15.5 deberta superglue leaderboard remove space let run example hugging face cloud platform run example run example hugging face

cloud platform click following link note stochastic nature transformer model produce different result run online platform continually change interface need accept adapt nlp task agnostic transformer models host inference api appear example output possible class name figure 15.6 deberta host inference api possible class name mobile website billing account access result interesting let compare gpt-3 keyword task sign https:// enter text input keyword ask engine find keyword text week upgrade ios version phone overheat use app keyword app overheating phone possible keyword app overheating phone go deberta gpt-3 transformer extend transformer vision model task agnostic models vision transformers foundation model see chapter 1 transformer distinct unique emergence transformer model qualify foundation model perform task train large model train supercomputer train learn specific task like model foundation model learn homogenization model domain fundamental architecture foundation model learn new skill datum fast well model gpt-3

google bert bert model train google task agnostic foundation model task agnostic model lead directly vit clip dall e model transformer uncanny sequence analysis ability level abstraction transformer model lead multi modal neuron multi modal neuron process image tokenize pixel image patch- e process word vision transformer image encode transformer model token word token show figure 15.7 figure 15.7 image encode word like token section vit vision transformer process image patch word clip vision transformer encode text image dall e vision transformer construct image text let begin explore vit vision transformer process image patch word vit vision transformers dosovitskiy et al 2021 sum essence vision transformer architecture de- sign title paper image worth 16x16 word transformer image recognition image convert patch 16x16 word nlp task agnostic transformer models let architecture vit look code basic architecture vit vision transformer process image patch word section process step split image patch 2 linear projection patch hybrid input embedding sublayer step split image equal sized patch step 1 split image patch image split n patch show figure 15.8 rule say patch long patch dimension 16x16 figure 15.8 split image patch patch equal dimension represent

word sequence problem patch remain type vision transformer image citation image cat section subsequent sec- tion take docchewbacca https://www.flickr.com/photos/st3f4n/ 2006 flickr free license https://creativecommons.org/ license sa/2.0/.

detail docchewbacca image flickr case vit step 2 linear projection flatten image step 2 linear projection flatten image step 1 convert image equal sized patch motivation patch avoid process- e image pixel pixel problem remain find way process patch team google research decide design linear projection flatten image patch obtain split image show figure 15.9 figure 15.9 linear projection flatten image idea obtain sequence work like patch remain problem embed sequence flatten image step 3 hybrid input embed sublayer word like image sequence fit transformer problem image google research decide hybrid input model job show figure 15.10 add convolutional network embed linear projection patch add positional encoding retain structure original image process embed input standard original bert like encoder nlp task agnostic transformer model figure 15.10 hybrid input sublayer standard encoder google research find clever way convert nlp transformer model vision transformer let implement hugging face example vision transformer code vision transformer code section focus main area code relate specific architecture open vision_transformers.ipynb github repository chapter google colab vm contain pre instal package torch torchvision display uncommente command cell notebook uncomment following command display list pre installed pip list -v vision transformer vit cell notebook notebook install hugging face transformer import necessary module pip install transformer transformer import vitfeatureextractor vitforimageclassification pil import image download image coco dataset find comprehensive corpus dataset website wish experiment https://cocodataset.org/ let download val2017 dataset follow coco dataset website instruction obtain image program download dataset locally val2017 contain 5,000 image choose test vit model run 5,000 image let test notebook image cat retrieve image cat url = http://images.cocodataset.org/val2017/000000039769.jpg image = image.open(requests.get(url stream = true).raw download

google feature extractor classification model feature_extractor = vitfeatureextractor.from_pretrained('google vit base- model = vitforimageclassification.from_pretrained('google vit base- model train 224 x 244 resolution image present 16 x 16 patch feature extraction classification notebook run model make prediction input = feature_extractor(images = image return_tensors="pt output = model(**inputs logit = outputs.logit note time write book hugging face warn code unstable constant evolution stop explore vit model test new territory cutting edge nlp task agnostic transformer models model predict 1000 imagenet class predicted_class_idx = logits.argmax(-1).item print("predicted class:",predicted_class_idx model.config output predict class 285 egyptian cat explore code follow prediction give information low level model.config.id2label list label class 1000 label class explain obtain class detailed text description 0 tench tinca tinca',1 goldfish carassius auratus 2 great white shark white shark man eater man eat shark carcharodon carcharias',3 tiger shark galeocerdo cuvieri', ,999 toilet tissue toilet paper bathroom tissue model display architecture model begin hybrid usage

convolutional input sublayer projection conv2d(3 768 kernel_size=(16 16 stride=(16 convolutional input embedding sublayer model bert like encoder time explore innovative nlp transformer transformer image lead transformer rapidly let clip computer vision model contrastive language image pre training clip follow philosophy transformer plug sequence datum transformer type layer instead send text pair time model send text image pair data tokenize encode embed clip task agnostic model learn text image pair sequence datum method contrastive look contrast feature image method use magazine game find difference contrast image let architecture clip look code basic architecture clip contrastive image train learn fit difference similarity image caption find way joint text image pretraining pretraine clip learn new task clips transferable learn new visual concept like gpt model action recognition video sequence caption lead endless application vit split image word like patch clip jointly train

text image encoder caption image pair maximize cosine similarity show figure 15.11 figure 15.11 jointly train text image figure 15.11 show transformer run standard transformer encoder text in- run resnet 50 layer cnn image transformer structure resnet 50 modify run average pooling layer attention pool mechanism multi head qkv attention head let clip learn text image sequence prediction clip code open vision_transformers.ipynb repository chapter github clip cell notebook nlp task agnostic transformer models program begin instal pytorch clip pip install ftfy regex tqdm pip install git+https://github.com/openai/clip.git program import module cifar-100 access image torchvision.dataset import cifar100 10,000 image available index 0 9,999 step select image want run prediction figure 15.12 select image index program load model device available gpu cpu load model device = cuda torch.cuda.is_available cpu model preprocess = clip.load('vit b/32 device image download download dataset cifar100 = cifar100(root = os.path.expanduser("~/.cache download = true input prepare prepare input image class_id = cifar100[index image_input = preprocess(image).unsqueeze(0).to(device text_input = torch.cat([clip.tokenize(f"a photo c c

let visualize input select run prediction import matplotlib.pyplot plt torchvision import transform output show index 15 lion figure 15.13 image index 15 know lion human transformer initially design nlp learn image recognize image program show run joint transformer model separate image input text input calculate feature calculate feature image_feature = model.encode_image(image_input text_feature = model.encode_text(text_inputs image section learn multiple layers features tiny images alex krizhevsky 2009 https://www.cs.toronto.edu/~kriz/learning- features-2009-tr.pdf cifar-10 cifar-100 dataset nlp task agnostic transformer models clip make prediction display prediction pick 5 similar label image image_feature /= image_features.norm(dim=-1 keepdim = true text_feature /= text_features.norm(dim=-1 keepdim = true similarity = 100.0 image_features text_features t).softmax(dim=-1 value index = similarity[0].topk(5 print result value index zip(values index print(f"{cifar100.classes[index]:>16s 100 value.item():.2f}% modify topk(5 want

obtain few prediction prediction clip find lion show flexibility transformer architecture cell display class class label class restrictive clip good job notebook contain cell describe architecture configuration clip explore model cell particularly interesting visual encoder begin convolutional embedding like vit model continue standard size-768 transformer multi head attention conv1 conv2d(3 768 kernel_size=(32 32 stride=(32 32 ln_pre layernorm((768 eps=1e-05 elementwise_affine = true ln_1 layernorm((768 eps=1e-05 elementwise_affine = true c_fc linear(in_features=768 out_features=3072 bias = true c_proj linear(in_features=3072 out_features=768 ln_2 layernorm((768 eps=1e-05 elementwise_affine = true interesting aspect model cell look size-512 text encoder run jointly image encoder ln_1 layernorm((512 eps=1e-05 elementwise_affine = true c_fc linear(in_features=512 out_features=2048 bias = true c_proj linear(in_features=2048 out_features=512 bias = true ln_2 layernorm((512 eps=1e-05 elementwise_affine = true

nlp task agnostic transformer models cell describe architecture configuration parameter clip represent datum show task agnostic transformer model process image text pair text text pair apply task agnostic model music text sound text music image type explore dall e task agnostic transformer model process image dall e clip task agnostic model clip process text image pair dall e pro- cesse text image token differently dall e input single stream text image 1,280 token 256 token text 1,024 token image dall e foundation model like clip dall e name salvador dali pixar wall e. usage dall e enter text prompt produce image dall e learn generate image text dall e 12 billion parameter version gpt-3 transformer generate image text description dataset text image pair basic architecture dall e unlike clip dall e concatenate 256 bpe encode text token 32×32 = 1,024 image token show figure 15.14 figure 15.14 dall e concatenate text image input figure 15.14 show time cat image concatenate input text dall e encoder decoder stack build hybrid architecture infuse convolutional function transformer model let peek code model work dall e code section dall e reconstruct image open vision_transformers.ipynb dall e cell notebook notebook install openai dall e pip install dall e notebook download image

process image import os sys import torchvision.transform t import torchvision.transforms.functional tf dall_e import map_pixel unmap_pixel load_model ipython.display import display display_markdown target_image_size = 256 resp = requests.get(url s = min(img.size s < target_image_size raise valueerror(f'min dim image s < target_image_size r = target_image_size s s = round(r img.size[1 round(r img.size[0 nlp task agnostic transformer models img = tf.resize(img s interpolation = pil.image lanczos img = tf.center_crop(img output_size=2 target_image_size img = torch.unsqueeze(t.totensor()(img 0 program load openai dall e encoder decoder change gpu e.g. cuda:0

dev = torch.device('cpu fast load time download file locally use local enc = load_model("https://cdn.openai.com dall e encoder.pkl dev dec = load_model("https://cdn.openai.com dall e decoder.pkl dev add enc dec cell look encoder decoder block hybrid model work convolutional functionality transformer model concatenation text image input image process section mycat.jpg creator denis rothman right reserve write permission require reproduce image chapter15 directory book repository download process finally display original image output display image figure 15.15 image cat program process display reconstruct image import torch.nn.functional f z_logit = enc(x z = torch.argmax(z_logits axis=1 z = f.one_hot(z num_classe = enc.vocab_size).permute(0 3 1 2).float x_stat = dec(z).float x_rec = unmap_pixels(torch.sigmoid(x_stats :3 x_rec = t.topilimage(mode='rgb')(x_rec[0 reconstruct image look extremely similar original figure 15.16

dall e reconstruct image cat result impressive dall e learn generate image dall e source code available time book writing openai api generate image text prompt online eye open meantime continue discover dall e openai https://openai.com/blog/ nlp task agnostic transformer models reach page scroll example provide example choose photo alamo square san francisco prompt figure 15.17 prompt alamo square sf modify night morning figure 15.18 modify

prompt dall e generate multitude text2image image figure 15.19 generate image text prompt implement vit clip dall e vision transformer let final thought finish expand universe model new transformer model like new smartphone emerge nearly week model mind blow challenging project manager ernie continual pretraine framework produce impressive result language challenge hug face provide model blow model baidu train exceed human baseline superglue leaderboard december 2021 https://super.gluebenchmark.com/leaderboard access good toy model purpose run automl small version model gain access baidu platform similar cost switch trillion parameter model optimize sparse modeling challenge paper fantastic model access real fully train model cost megatron turing 500 billion parameter transformer model challenge good model market access api blow model cost xlnet pretraine like bert author contend exceed bert model perfor- challenge xlnet exceed performance google bert version google use activity access good version google bert xlnet model list endless grow test remain challenge issue mention previously trans- model qualify foundation model foundation model fully train wide range task able perform task train unique level nlu sufficiently large guarantee reasonably accurate result openai gpt-3 nlp task agnostic transformer model site offer transformer prove useful educational purpose consider sufficiently trained large qualify benchmarking good approach deepen understanding transformer model possible point expert find way jungle big tech innovation easy choose smartphone new transformer model appear market good practice cutting edge research read publication book test system lead assess transformer model

research read publication book test system lead assess transformer model choose implement spend month explore model appear market change mod- el month project production industry 4.0 move seamless api ecosystem learn model impossible understand new model quickly achieve deepen knowledge transformer model basic structure transformer model remain unchanged layer encoder and/or decoder stack remain identical attention head parallelize optimize computation reformer model apply lsh bucket chunking recompute layer

input instead store information optimize memory issue

billion parameter model gpt-3 produce acceptable result example deberta model disentangle content position make training process flex- ible result impressive billion parameter model gpt-3 equal output deberta vit clip dall e take fascinating world task agnostic text image vision transformer model combine language image produce new productive information question remain far ready use ai automate system at- tempt visualize future transformer base ai chapter rise metahuman reformer transformer model contain encoder true false reformer transformer model contain decoder true false input store layer layer reformer model true false deberta transformer model disentangle content position true false necessary test hundred pretrained transformer model choose project true false late transformer model good true false well transformer model nlp task multi task transformer 8 transformer model need fine tune true false openai gpt-3 engine perform wide range nlp task fine tuning true/ 10 well implement ai algorithm local server true false hugging face reformer https://huggingface.co/transformers/model_doc/reformer hugging face deberta https://huggingface.co/transformers/model_doc/deberta pengcheng xiaodong liu jianfeng gao weizhu chen 2020 decoding enhance bert disentangle attention https://arxiv.org/abs/2006.03654 alexey dosovitskiy lucas beyer alexander kolesnikov dirk weissenborn xiaohua zhai thomas unterthiner mostafa dehghani matthias minderer georg heigold sylvain gelly jakob uszko- reit neil houlsby 2020 image worth 16x16 word transformers image recognition scale https://arxiv.org/abs/2010.11929 william fedus barret zoph noam shazeer 2021 switch transformer scale trillion parameter model simple efficient sparsity https://arxiv.org/abs/2101.03961 alec radford jong wook kim chris hallacy aditya ramesh gabriel goh sandhini agarwal girish sastry amanda askell pamela mishkin jack clark gretchen krueger ilya sutskever 2021 learn transferable visual model natural language supervision https:// nlp task agnostic transformer model aditya ramesh mikhail pavlov gabriel goh scott gray chelsea voss alec radford mark chen ilya sutskever 2021 zero shot text image generation

https://arxiv.org/

join book discord space join book discord workspace monthly ask session author emergence transformer- industry 4.0 i4.0 reach maturity machine machine connection communication decision making ai primarily embed ready use pay as- cloud ai solution big tech absorb talented ai specialist create api interface integration tool ai specialist development design architect integrator cloud ai pipeline administrator ai job engineer consultant engi- chapter 1 transformer introduce foundation model transformer nlp task train chapter 15 nlp task agnostic transformer models expand- e foundation model transformer task agnostic model perform vision task nlp task chapter extend task agnostic openai gpt-3 model wide range copilot task new generation ai specialist datum scientist learn work ai copilot help generate source code automatically decision chapter begin explore prompt engineering detail example task consist convert meeting note summary transformer boost productivity natural language remain challenge ai emergence transformer drive copilot learn use openai codex copilot github copilot suggest source code write program codex codex convert natural language code discover new ai method domain specific gpt-3 engine chapter generate embedding 12,288 dimension plug machine learn- e algorithm ask transformer produce instruction automatically filter bias input output look transformer drive recom- mender ai 2020s build ethical method recommender system permeate social medium platform suggest video post mes- sage book product want consume build educational multi purpose transformer base recommender system ml process transformer model analyze sequence begin nlp successfully expand computer vision explore transformer base computer vision program develop jax finally ai copilot contribute transition virtual system metaverse expand decade pilot develop application howev- er code develop activate completion limit method line code ide suggest list method copilot produce completion paragraph code chapter cover following topic codex language source code model embed drive machine learning content filter model explore transformer base recommender extend nlp sequence learning behavior prediction implement

transformer base recommender extend nlp sequence learning behavior prediction implement transformer model jax apply transformer model computer vision let begin prompt engineering critical ability acquire speak specific language hereditary language center brain contain- e language parent brain engineer neuron early life speak read write understand language human different language circuitry depend cultural background communicate early year grow discover hear chaos unfinished sentence grammar mistake misused word bad pronunciation distortion use language convey message quickly find need adapt language person audience address try additional input prompt obtain result output expect foundation level transformer model gpt-3 perform hundred task indefinite number way learn language transformer prompt response language effective communication person near human level transformer contain minimum information maximize result represent minimum input information obtain result mini maximum output system maxr. represent chain communication replace input prompt transformer input influence model react output response dialogue transformer d(t mini(prompt 1 probability maxr(response 1 mini(prompt 0 probability maxr(response 0 quality d(t depend

define mini(prompt prompt tend reach 1 produce probability tend 1 prompt tend reach 0 produce output probability tend 0 prompt content impact probability transformer include prompt response estimation emergence transformer drive copilot take year learn language child adult take time learn language transformer design mini(prompt effectively need understand architecture way algorithm calculate prediction need spend time understand design input prompt transformer behave expect section focus oral language prompt openai gpt-3 nlp task take meeting note conversation tend unstructured transform meeting note conversation summary challenging section focus summarize note conversation seven situation casual english casual formal english limited context begin casual english meaningful context casual english meaningful context casual english speak short sentence limited

vocabulary let ask openai gpt-3 perform note summary task www.openai.com log sign example page select note summary gpt-3 information require summarize casual conversation jane tom jane tom developer start work tom offer jane coffee jane decline case mini(prompt)=1 input information fine show figure 16.1 figure 16.1 summarize document note click generate surprisingly good answer show figure 16.2 figure 16.2 acceptable summary provide gpt-3 conclude ai find structure chaotic daily conversation meeting aimless chatter answer easy complicate input add metonymy emergence transformer driven copilots casual english metonymy tom mention word coffee set gpt-3

track tom word java instead coffee coffee refer beverage java ingredient come island java metonymy use attribute object java coffee java programming language logo cup coffee face possible definition java ingredient coffee mean coffee meton- ymy island java programming language gpt-3 polysemy meaning word issue solve human master polysemy learn different meaning word know word mean context case jane tom developer complicate situation talk coffee language answer easy human tom talk wife stop drink gpt-3 confuse polysemy word java replace coffee produce incorrect answer figure 16.3 incorrect gpt-3 response prompt confusing confirm mini(prompt 0 probability maxr(response 0 human conversation difficult analyze add ellipsis casual english ellipsis situation bad let suppose tom drink cup coffee jane look cup coffee casually greet instead ask jane want coffee java tom say tom leave word coffee ellipsis jane understand tom mean look hold cup coffee openai gpt-3 detect word drinking manage associate verb question want want programming language following summary produce gpt-3 correct figure 16.4 correct response produce gpt-3 let happen vague context human understand re- main challenge ai casual english vague context tom need mention wife jane understand talk hold cup coffee let remove tom reference wife verb drinking let leave want instead coffee java figure 16.5 vague input context output reflect apparent chaos conversation figure 16.6 poor gpt-3 response emergence transformer drive copilot prompt vague lead inadequate response sum d(t 0 mini(prompt 0 probability maxr(response 0

human communicate bring culture past relationship visual situation invisible factor conversation invisible factor party body language etc listen people refer thing know movie sport prob- lem factory etc cultural event culture different list endless invisible factor ai blind let introduce sensor situation casual english sensor introduce video sensor room thought experiment imagine use image captioning video feed supply context early dialogue human generate dialogue people know understand consider follow dialogue jane tom video feed produce image captioning show tom drink cup coffee jane type keyboard jane tom developer mumble way day get work open space provide following chaotic chat prompt tom hi jane yeah sure tom want jane nope tom cool try jane yup tom sleep well jane yeah sure output gpt-3 acceptable important semantic word miss start summarize developer see type keyboard developer enter room offer cup coffee decline insist chat sleep coffee result change run gpt-3 look probability select good gpt-3 experiment image captioning provide tom hold cup coffee deprive gpt-3 visual context casual english sensor visible context difficult situation ai tom refer event day today suppose tom come cup coffee morning come ask jane want get coffee thought experiment imagine possible case case video feed thought experiment reveal chaos video feed developer accountant consultant let context leave following context let dialogue contain tom jane need mention context leave tom hi jane yeah sure tom want jane nope tom cool try jane yup tom sleep well jane yeah sure output astonishing casual language jane tom lead gpt-3 absurd conclusion remember gpt-3 stochastic algorithm slight change input lead different output gpt-3 try guess talk gpt-3 detect conversation consume casual language lead nonsensical prediction illegal substance reproduce section ethical reason gpt-3 determine level language associate related situation happen reproduce experiment formal english formal english conversation context let context provide formal

formal english formal english conversation context let context provide formal english formal english contain long sentence good grammar manner express conversation contain context

formal english tom good morning jane jane good morning tom tom want jane thank fine tom excellent right track jane yes tom sleep well day jane yes thank gpt-3 naturally understand tom refer drinking level english good manner output satisfactory summarize tom say good morning jane tom offer drink jane say thank fine tom say excellent right track jane say yes tom ask sleep well day emergence transformer drive copilot imagine endless number variation conversation introduce people dialogue object generate endless number situation let sum experiment prompt engineering training thought chaotic human use method reconstruct unstructured sentence human need ask additional question understand somebody talk need accept interact train transformer openai gpt-3 mind dialogue d(t transformer response maxr(response depend quality input mini(prompt define beginning section mini(prompt 1 probability maxr(response 1 mini(prompt 0 probability maxr(response 0

practice prompt engineering measure progress time prompt engineering new skill level ai prompt engineering ability lead able master copilot welcome world ai drive development copilot power openai available let begin github copilot section use github copilot pycharm jetbrains follow instruction documentation install jetbrains activate openai github copilot pycharm work withgithub copilot step process figure 16.7 openai codex train public code text internet train model plug github copilot service github service manage forth flow code write editor case pycharm openai codex github service manager make suggestion send interaction improvement code editor development workspace figure 16.7 github copilot step process follow instruction provide github copilot log github pycharm troubleshooting read https://copilot.github.com/#faqs set pycharm editor simply type import matplotlib.pyplot plt emergence transformer drive copilot soon code type open openai github suggestion pane sug- figure 16.8 suggestion code type choose copilot suggestion prefer appear editor confirm suggestion tab key wait suggestion draw scatterplot import matplotlib.pyplot plt def draw_scatterplot(x y draw_scatterplot([1 2 3 4 5 1 4 9 16 25

plot display figure 16.9 github copilot scatterplot run result machine github_copilot.py chapter16 folder book github repository technology seamless invisible progressively expand area development system pack gpt-3 functionality pipeline technology available python javascript training prompt engineering work github copilot drive openai codex let directly openai codex good place train copilot openai codex suggest source code codex translate natural language openai website click link codex interface provide instruction result display codex window source code right pane interface emergence transformer drive copilot example section contain sequence prompt pane figure 16.10 codex javascript sandbox enter natural language provide instruction pane generate javascript run example enter instruction natural language draw 50 small ping pong ball sort color ball round codex derive gpt-3 case engine refer davinci codex mean codex inherit property gpt-3 engine understand let parameter top_p engine sample output engine example top_p set 0.1 10 sampling account retrieve token 10 probability mass computation stochastic choice set choice run patient time design prompt ai engine learning path learn understand behavior codex engine 2 accept free creative nature stochastic algorithm pilot well prompt able grow codex engine improve script appear javascript pane comment instruction code draw 50 small ping pong ball sort color var ball = var = 0 < 50 i++ var ball = document.createelement('div ball.style.width = 10px ball.style.height = 10px ball.style.backgroundcolor = hsl + math.random 360 + 100 ball.style.position = absolute ball.style.left = math.random window.innerwidth + px ball.style.top = math.random window.innerheight + px ball round var = 0 < balls.length i++ balls[i].style.borderradius = 50 result display figure 16.11 create multicolored ball emergence transformer drive copilot let ask program ball ball inside window code generate ball inside window var moveball = function var = 0 < balls.length i++ var ball = balls[i ball.style.left = parseint(ball.style.left + math.random 10 5 ball.style.top = parseint(ball.style.top + math.random

10 5 + ball export code html click export jsfiddle button figure 16.12 export jsfiddle button jsfiddle create html page figure 16.13 jsfiddle create html page case code save codex.html

chapter folder github repository book open watch innovative result create html page language natural code source code domain specific gpt-3 engine section explore gpt-3 engine perform domain specific task run model subsection section embedding2ml use gpt-3 provide embedding ml algorithm instruct series ask gpt-3 provide instruction task content filter filter bias form unacceptable input output begin embedding2ml embedding input ml openai train embed model different dimension different capa- ada 1,024 dimension babbage 2,048 dimension curie 4,096 dimension davinci 12,288 dimension explanation engine find information openai website davinci model offer embedding 12,288 dimension section use power davinci generate embedding supply chain dataset send embedding embedding sublayer transformer send embedding cluster machine learning program scikit learn library step step 1 instal import openai enter api key step 2 load dataset step 3 combine column emergence transformer drive copilot step 4 run gpt-3 embedding step 5 cluster k means embedding step 6 visualize cluster t sne process sum figure 16.14 figure 16.14 step process send embedding clustering algorithm open google colab file domain_specific_gpt_3_functionality.ipynb embedding2ml gpt-3 engine section notebook step describe section match notebook cell let summary step process step 1 instal import openai let start following substep run cell restart runtime run cell sure restart runtime pip install openai

enter api key load dataset step 2 load dataset load file run cell upload tracking.csv available github repository book contain scm datum import pandas pd df = pd.read_csv('tracking.csv index_col=0 data contain seven field let print line following command time product user score summary text 1 01/01/2016 06:30 wh001 c001 4 time agv1 2 01/01/2016 06:30 wh001 c001 8 late r1 nan 3 01/01/2016 06:30 wh001 c001 2 early r15 nan 4 01/01/2016 06:30 wh001 c001 10 deliver r20 nan 5 01/01/2016 06:30 wh001 c001 1 time r3 nan 1049 01/01/2016 06:30 wh003 c002 9 time agv5 nan 1050 01/01/2016 06:30 wh003 c002 2 late agv10 nan 1051 01/01/2016 06:30 wh003 c002 1 early agv5 nan 1052 01/01/2016 06:30 wh003 c002 6 deliver agv2 nan 1053 01/01/2016 06:30 wh003 c002 3 time agv2 nan 1053 row x 7 column emergence

transformer drive copilot combine column build cluster wish step 3 combine column combine product column summary obtain view product deliv- ery status remember experimental exercise real life project carefully analyze decide column wish combine following example code replace choice experimentation df['combined = df summary.str.strip()+ + df.product.str.strip new column name combine time product user text combine 1 01/01/2016 06:30 wh001 c001 agv1 time wh001 2 01/01/2016 06:30 wh001 c001 r1 nan late wh001 3 01/01/2016 06:30 wh001 c001 r15 nan early wh001 4 01/01/2016 06:30 wh001 c001 r20 nan deliver wh001 5 01/01/2016 06:30 wh001 c001 r3 nan time wh001 1049 01/01/2016 06:30 wh003 c002 agv5 nan time wh003 1050 01/01/2016 06:30 wh003 c002 agv10 nan late wh003 1051 01/01/2016 06:30 wh003 c002 agv5 nan early wh003 1052 01/01/2016 06:30 wh003 c002 agv2 nan deliver wh003 1053 01/01/2016 06:30 wh003 c002 agv2 nan time wh003 1053 row x 8

column run embedding model combine column step 4 run gpt-3 embedding run davinci similarity model obtain 12,288 dimension combine start time start = time.time def get_embedding(text engine="davinci similarity text = text.replace("\n return openai engine(id = engine).embeddings(input = text])['data'][0 df['davinci_similarity = df.combined.apply(lambda x get_embedding(x end time end = time.time conversion = datetime.timedelta(seconds = etime result impressive 12,288 dimension combine column time davinci 1 01/01/2016 06:30 -0.0047378824 0.011997132 -0.017249448 2 01/01/2016 06:30 -0.009643857 0.0031537763 -0.012862709 3 01/01/2016 06:30 -0.0077407444 0.0035147679 -0.014401976 4 01/01/2016 06:30 -0.007547746 0.013380095 -0.018411927 5 01/01/2016 06:30 -0.0047378824 0.011997132 -0.017249448 1049 01/01/2016 06:30 -0.0027823148 0.013289047 -0.014368941 1050 01/01/2016 06:30 -0.0071367626 0.0046446105 -0.010336877 emergence transformer drive copilot 1051 01/01/2016 06:30 -0.0050991694 0.006131069 -0.0138306245 1052 01/01/2016 06:30 -0.0066779135 0.014575769 -0.017257102 1053 01/01/2016 06:30 -0.0027823148 0.013289047 -0.014368941 1053 row x 9 column need convert result numpy

matrix create matrix import numpy np matrix = np.vstack(df.davinci_similarity.values matrix shape 1,053 record x 12,288 dimension impressive matrix ready send scikit learn machine learning clustering algorithm step 5 cluster k mean clustering embedding usually send classical dataset k mean clustering algorithm send 12,288 di- mension dataset ml algorithm sublayer transformer import k means scikit learn sklearn.cluster import kmeans run classical k mean clustering algorithm 12,288 dimension dataset n_cluster = 4 kmean = kmeans(n_cluster = n_cluster init='k means++',random_state=42 label = kmeans.label df['cluster = label output cluster request print label content dataset output 2 3 0 0 1 2 let visualize cluster t sne step 6 visualize cluster t sne t sne keep local similarity pca maximize large pairwise distance case small pair- notebook use matplotlib display t sne sklearn.manifold import tsne import matplotlib.pyplot plt visualize need run t sne algorithm tsne = tsne(n_components=2 perplexity=15 random_state=42 init='random vis_dims2 = tsne.fit_transform(matrix display result matplotlib x = x x y vis_dims2 y = y x y vis_dims2 category color enumerate(['purple

green red blue xs = np.array(x)[df cluster==category ys = np.array(y)[df cluster==category emergence transformer drive copilots plt.scatter(xs ys color = color alpha=0.3 avg_x = xs.mean avg_y = ys.mean plt.scatter(avg_x avg_y marker='x color = color s=100 plt.title("clusters embedding t sne plot show cluster data point pile datum point circle cluster attach close centroid figure 16.15 cluster embedding t sne run large gpt-3 model embed 12,288 dimension plug result clustering algorithm potential combine transformer machine learning endless peek embedding section notebook wish peek data frame let look instruct series personal assistant avatar metaverse website domain increasingly need provide clear instruction user ask help instruct series section section ask transformer explain set parent control microsoft edge following prompt explain set parent control edge run completion cell openai.api_key = os.getenv("openai_api_key response = openai.completion.create prompt="explain set parent control edge.\n\n\nactions r = response["choices"][0 response list instruction request 1 start internet explorer 2 click tool menu 3 click internet option 4 click advanced tab 5 click clear

select enable personalized favorite menu check number instruction ask unlimited use creativity imagination find example input output acceptable let implement content filter bias unacceptable language form unethical input exclude ai emergence transformer drive copilot openai train model content filter run example section content filter section domain_specific_gpt_3_functionality.ipynb recommendation filter input output show figure 16.16 figure 16.16

implement content filter recommendation implement step process apply content filter input datum let ai algorithm run train apply content filter output datum section input output datum name content obnoxious input following content = small fat child play basketball school input unacceptable school nba basketball remain nice exercise let run content filter cell content filter alpha response = openai.completion.create en prompt = < |endoftext|>"+content+"\n--\nlabel content filter store result response dictionary object retrieve value choice obtain level acceptability r = response["choices"][0 print("content filter level r["text content filter send value 0 safe 1

sensitive 2 unsafe case result 2 course content filter level 2 content filter sufficient recommend add algorithm control filter input output content rule basis dictionary method explore domain specific model let build transformer base recommender transformer base recommender system transformer model learn sequence learn language sequence great place start consi- dere billion message post social medium cloud platform day consumer behavior image sound represent sequence section create general purpose sequence graph build gener- al purpose transformer base recommender google colaboratory deploy metahuman let define general purpose sequence activity represent entity link organize sequence example video youtube entity link behavior person go video video e. emergence transformer drive copilot example bad fever entity f link inference doctor lead micro decision b. purchase product d amazon consumer generate link suggestion c product example infinite define entity section

letter speak language follow grammar rule escape example suppose a= e= eat d= candy proper sequence express fact consume candy eat candy somebody say eat candy sound slightly sequence link represent rule a->e eat automatically infer rule domain observe behavior learn dataset ml manually listen expert section suppose observe youtube user month spend hour watch video notice user systematically go type video example video singer b video singer d. behavior rule x person p represent system entity vertex graph link edge example apply x(p vertex obtain following undirected graph figure 16.17 graph youtube user video combination suppose vertex video viewer favorite singer c singer viewer prefer value 1 statistical transition link edge viewer past week value 100 viewer favorite singer video viewer path represent edge vertex value v(r(p goal recommender suggest sequence lead video singer c suggest c directly case represent undirected graph reward matrix r r = ql.matrix 0,0,0,0,1,0 let use reward matrix simulate activity viewer x month dataset pipeline simulation rl mdp section

activity viewer x month dataset pipeline simulation rl mdp section simulate behavior x person p watch video song youtube define x(p determine value behavior p v(x(p organize value reward matrix r markov decision process mdp implement bellman equation open kantaibert_recommender.ipynb

chapter folder book github re- pository notebook modification kantaibert.ipynb describe chapter 4 pretraine roberta model scratch chapter 4 train transformer kant.txt contain work immanuel kant section generate thousand sequence person behavior reinforcement learning rl rl scope book section con- tain reminder step train transformer model learn simulate person behavior emergence transformer driven copilots train customer behavior mdp kantaibert.ipynb chapter 4 begin load kant.txt train roberta distilbert architecture kant.txt contain work immanuel kant section generate se- quence reward matrix r define general purpose sequence section chapter r = ql.matrix 0,0,0,0,1,0 cell program step 1a training

dataset pipeline simulation rl mdp cell implement mdp bellman equation bellman mdp base q function q[current_state action = r[current_state action + gamma maxvalue equation r original reward matrix q update matrix size r. update rein- forcement learning compute relative value link edge entity gamma learning rate set 0.8 avoid overfitte training process maxvalue maximum value vertex example viewer p you- tube video view singer program increase value e suggestion appear recommendation little little program try find good value help viewer find good video watch reinforcement program learn good link edge recommend good viewing sequence original reward matrix train operational matrix add original entity train value clearly appear b c d e f 0 0 0 0 258.44 0 0 0 0 321.8 0 207.752 b 0 0 500 321.8 0 0 c 0 258.44 401 0 258.44 0 d 207.752 0 0 321.8 0 0 e 0 258.44 0 0 0 0 f original sequence value v behavior x person p train v(x(p))={ae=259.44 bd=321.8 bf=207.752 c=500 cd=321.8 db=258.44 de=258.44 ea=207.752 ed=321.8 fb=258.44

change possible recommend sequence exciting video p preferred singer suppose p view video singer e. line e train matrix recommend video high value line d=321.8 video singer d appear youtube feed person p. goal section stop phase instead section use mdp create meaningful sequence create dataset transformer use training youtube need generate sequence create dataset youtube store behavior viewer big datum google powerful algorithm recommend good video video feed viewer platform use cosine similarity implement chapter 9 matching tokenizers data- set prediction mdp train youtube viewer amazon buyer google search result doctor diagnosis path supply chain type sequence transformer take sequence learning training prediction level emergence transformer drive copilot let implement simulation create behavior sequence transformer model simulate consumer behavior mdp rl program train cell 1 cell 2 step 1b apply dataset pipeline simulation mdp simulate youtube viewer behavior month include similar viewer profile add simulation 10,000 sequence video watching cell 2 begin create kant.txt file train kantaibert transformer simulate decision making process f = open("kant.txt w entity vertex introduce number sequence set 10,000 function choose random

start vertex name origin program use train matrix select good sequence domain point origin case suppose favorite singer person following fbdc edc edc dc bdc aedc aedc bdc bdc aedc bdc aedc edc bdc aedc dc aedc 10,000 sequence calculate kant.txt contain dataset transformer kant.txt remain cell program kantaibert.ipynb describe chapter 4 pretraine roberta model scratch transformer ready recommendation chapter 4

kantaibert.ipynb contain following mask sequence fill_mask("human thinking involve human < mask > sequence specific relate immanuel kant work notebook general pur- pose dataset domain notebook input output contain duplicate cleaning function filter obtain non du- sequence bdc fbdc token_str fbdc sequence bdc dc token_str dc sequence sense viewer watch video behavior chaotic machine learning come ai sequence generate convert natural language user interface metahuman section refer recommender take large amount feature exceed human capacity reason parameter lead accurate prediction human pragmatic metahumans digital human context powerful comput- e tool metahumans digital sense humans ai copilot example bdc sequence song singer b follow singer d p favorite singer c. emergence transformer drive copilot sequence convert natural language option possible sequence send bot digital human use metahuman educational video wait api metahuman insert interface voice message example google maps car listen voice sound like human slip think person machine invisible embed suggestion amazon remain make recommendation lead micro decision influence sales- person invisible metahuman case general purpose sequence create mdp train roberta transformer show transformer apply type sequence let transformer apply computer vision book nlp computer vision previous section implement general purpose sequence apply domain computer vision title article dosovitskiy et al 2021 say image worth 16x16 word trans- former image recognition scale author process image sequence result prove point google vision transformer available colaboratory notebook open vision transformer_mlp_mixer.ipynb chapter16 directory book github repository open vision_transformer_mlp_mixer.ipynb contain transformer computer vision model jax jax

combine autograd xla jax differentiate python numpy function jax speed python numpy compilation technique parallelization emerge technology appear jump train ride know technology evolve google metahuman platform case remain cutting edge learn circumvent limit find way use new technology notebook self explanatory explore work bear mind industry 4.0 reach maturity industry 5.0

kick good implementation obtain integrate datum cloud ai platform local development diminish company turn cloud ai bear local development maintenance support notebook table content contain transformer process go time book time simply apply sequence digital image infor- figure 16.18 vision transformer notebook notebook follow standard deep learning method show image label image label image label = batch['image'][0][:9 batch['label'][0][:9 title = map(make_label_getter(dataset labels.argmax(axis=1 emergence transformer drive copilot figure 16.19 image label notebook contain standard transformer process display training image train image note image crop scale differently check input_pipeline.get_data editor right image preprocesse differently batch = next(iter(ds_train.as_numpy_iterator image label = batch['image'][0][:9 batch['label'][0][:9 title = map(make_label_getter(dataset labels.argmax(axis=1 image chapter learn multiple layer feature tiny images

alex krizhevsky 2009 https://www.cs.toronto.edu/~kriz/learning- features-2009-tr.pdf cifar-10 cifar-100 dataset figure 16.20 train datum transformer program classify random picture like miracle transformer model originally design nlp use general purpose sequence recommender computer vision begin explore generalization simplicity model surprising vision transformer rely architecture transformer contain complexity convolutional neural network pro- duce comparable result robot bot equip transformer model understand language interpret image understand world vision transformer implement metahuman metaverse human ai copilot metaverse humans metahuman ai merge metaverse explore metaverse scope book toolbox provide book show path metaverse populate human metahuman ai emergence transformer

driven copilots avatars computer vision video game experience communication immersive look smartphone location look evolution look natural invent computer add screen invent smartphone use app video meeting enter virtual reality type meeting activity use facebook metaverse example smartphone feel present location people personal professional meet feel present doubt major evolution smartphone communication feel present different look small screen mobile metaverse impossible possible spacewalk surf huge wave walk forest visit dinosaur imagination take yes limit danger threat go human technology use ai control ai see content filtering transformer tool book add emerge metaverse technology literally world good use knowledge skill acquire book create ethical future metaverse physical world chapter describe rise ai copilot human decision making level capability in- dustry 4.0 open door machine interconnectivity machine machine micro deci- sion making speed transaction ai copilot boost productivity wide range see use openai codex generate source code code natural build transformer base recommender system dataset generate mdp pro- gram train roberta transformer model dataset structure multi purpose sequence model metahuman acquire multi domain recommender functionality chapter show vision transformer classify image process sequence finally see metaverse recommendation visible metahuman interface invisible deeply embed function social medium example transformer emerge innovate copilot model incredibly complex new era journey prove challenging exciting

copilot model incredibly complex new era journey prove challenging exciting ai copilot generate code automatically exist true false 2 ai copilot replace human true false gpt-3 engine task true false transformer train recommender true false transformer process language true false 6 transformer sequence contain word true false vision transformer equal cnn true false 8 ai robot computer vision exist true false impossible produce python source code automatically true false 10 day copilot robot true false openai platform gpt-3 https://openai.com openai model engine https://beta.openai.com/docs/engines vision transformers alexey dosovitskiy lucas beyer alexander kolesnikov dirk weissenborn xiaohua zhai thomas unterthiner mostafa

dehghani matthias minderer georg heigold syl- vain gelly jakob uszkoreit neil houlsby 2020 image worth 16x16 word transformers image recognition scale https://arxiv.org/abs/2010.11929 jax vision transformer https://github.com/google/jax openai visual studio copilot https://copilot.github.com/ facebook metaverse https://www.facebook.com/meta/videos/577658430179350 emergence transformer drive copilots markov decision process mdp example graph denis rothman 2020 artificial intelligence example 2 nd edition https://www.amazon.com/artificial-intelligence- join book discord space join book discord workspace monthly ask session author appendix terminology past decade produce convolutional neural networks cnns recurrent neural networks rnns type artificial neural networks anns certain vocabulary common transformer model introduce new word existing word slightly differently appendix briefly describe transformer model clarify usage deep learning vocabulary apply transformer motivation transformer model architecture rely industrial approach deep learning geometric nature transformer boost parallel processing addition ar- chitecture transformer perfectly fit hardware optimization requirement google exam- ple take advantage stack structure transformer design domain specific optimize hardware require float number precision design transformer model imply take hardware account architecture transformer combine software hardware optimization start appendix define new usage neural network language terminology transformer model stack contain identically size layer differ classical deep learning model show figure i.1 stack run stack encoder decoder figure i.1 layer form stack transformer stack learn rise stack layer

i.1 layer form stack transformer stack learn rise stack layer transmit learn layer memory imagine stack empire state building new york city far ascend throught office high floor look window finally fantastic view manhattan layer contain sublayer show figure i.2 sublayer different layer identical structure boost hardware optimization original transformer contain sublayer run self attention sublayer design specifically nlp hardware optimization classical feedforward network tweaking figure i.2 layer contain sublayer self attention sublayer divide n

independent identical layer call head ex- ample original transformer contain head figure i.3 represent head processor transformer industrialized structure fit figure i.3 self attention sublayer contain head note attention head represent microprocessor figure i.3 stress parallel processing power transformer architecture transformer architecture fit nlp hardware optimization requirement terminology transformer models join book discord space join book discord workspace monthly ask session author appendix ii hardware constraint transformer transformer model exist optimize hardware memory disk management design remain critical component computing power remain prerequisite nearly impossible train original transformer describe chapter 2 get start architecture transformer model gpu gpu center battle efficient transformer model appendix chapter 3 fine tune bert models importance gpu step architecture scale transformer cpu versus gpu implement gpu pytorch example optimize language architecture scale transformer hint hardware drive design appear architecture multi head attention section chapter 2 getting start architecture transformer model point view time analyze sequence dmodel block furthermore calculation time find perspective well way divide dmodel = 512 dimension word xn x word sequence 8 dk = 64 dimension hardware constraint transformer model run 8 head parallel speed training obtain 8 different repre- sentation subspace word relate figure ii.1 multi head representation 8 head run parallel easily motivation force attention head learn 8 different perspective dig deeply motivation original 8 attention head perform dif- ferent calculation parallel lead directly hardware

attention head perform dif- ferent calculation parallel lead directly hardware optimization brown et al 2020 language models shot learners https://arxiv.org/abs/2005.14165 describe design gpt model confirm transformer architecture hard- partition model gpu depth width dimension minimize data transfer node precise architectural parameter model choose base computational efficiency load balancing layout model gpu transformer differ construction encoder decoder size hardware constraint require parallel processing need step gpu special gpu special clue gpu drive design emerge architecture multi head attention section chapter 2

getting start architecture transformer model attention define scale dot product attention represent follow equation plug q k v (=

conclude following attention head design parallel computing attention head base matmul matrix multiplication gpu design parallel computing cpu central processing unit optimize serial processing run attention head serial processing far long train efficient transformer model small educational transformer run cpu qualify state of- gpu graphic processing unit design parallel processing transformer model design parallel processing gpu serial processing cpu gpu design matrix multiplication nvidia gpu example contain tensor core accelerate matrix operation significant proportion artificial intelligence algorithm use matrix operation include transformer model nvidia gpu contain goldmine hardware optimization matrix operation follow link

provide information google tensor processing unit tpu equivalent nvidia gpu tensorflow optimize use tensor tpus tpu https://cloud.google.com/tpu/docs/tpus tensor tensorflow https://www.tensorflow.org/guide/tensor bertbase 110 m parameter initially train 16 tpu chip bertlarge 340 m param- eter train 64 tpu chip train bert https://arxiv.org/ establish architecture transformer perfectly fit constraint par- allel hardware need address issue implement source code run gpu hardware constraint transformer models implement gpu code pytorch language framework manage gpu pytorch contain tensor tensorflow tensor look like numpy np.array numpy fit parallel processing tensor use parallel processing feature gpu tensor open door distribute datum gpu pytorch framework chapter03 notebook bert_fine_tuning_sentence_classification_gpu.ipynb cuda compute unified device architecture communicate nvidia gpu cuda nvidia platform general computing gpu specific instruction add source code https://developer.nvidia.com/cuda-zone chapter03 notebook cuda instruction transfer model datum nvidia gpu pytorch instruction specify device wish use torch.device https://pytorch.org/docs/stable/notes/cuda.html explain device illustrate implementation gpu pytorch program general let focus select device datum parallelism load model device add

batch datum device bullet point contain way device cell number select device cell 3 program check cuda available nvidia gpu device device = torch.device("cuda torch.cuda.is_available data parallelism cell 16 model distribute parallel computing gpu gpu available model = bertforsequenceclassification.from_pretrained("bert base- model = nn dataparallel(model load model device cell 16 model send device add batch device cell 20 training validation datum batch datum add gpu available 1 n add batch gpu batch = tuple(t.to(device t batch follow section

describe test illustrate use gpu transformer model run notebook chapter runtime configuration test gpu google colab section describe informal test run illustrate potential gpu use chapter03 notebook bert_fine_tuning_sentence_classification_gpu.ipynb run notebook scenario google colab free cpu google colab free gpu google colab pro google colab free cpu nearly impossible fine tune train transformer model million billion pa- rameter cpu cpu sequential transformer model design parallel runtime menu change runtime type submenu select hardware acceler- ator cpu gpu tpu hardware constraint transformer model test run cpu show figure ii.2 figure ii.2 select hardware accelerator notebook reach training loop slow right start figure ii.3 training loop 15 minute happen cpu design parallel processing transformer model design parallel processing toy model require gpu google colab free gpu let notebook setting select gpu figure ii.4 select gpu time write test google colab nvidia k80 attribute vm figure ii.5 nvidia k80 gpu activate training loop advance normally last 20 minute google colab vms time test november 2021 provide gpu gpu expen- sive case figure ii.6 show training loop perform reasonable time figure ii.6 training loop k80 gpu hardware constraints transformer models find interesting google colab pro provide fast gpu google colab pro gpu vm activate google colab provide nvidia p100 gpu show figure ii.7 interesting original transformer train 8 nvidia p100s state vaswani et al.(2017 attention need take 12 hour train base model 10 parameter 8 gpu figure ii.7 google colab pro vm provide p100 gpu training loop time considerably reduce last 10 minute show figure ii.8 training loop p100 gpu join book

discord space join book discord workspace monthly ask session

author appendix iii generic text completion gpt-2 appendix detailed explanation generic text completion gpt-2 section chapter 7 rise suprahuman transformer gpt-3 engine section describe implement gpt-2 transformer model generic text complexion read usage notebook directly chapter 7 build program run appendix profound knowledge gpt model work clone openai_gpt_2 repository download 345m parameter gpt-2 transformer model interact enter context sentence analyze text generate transformer goal create new content section divide step open openai_gpt_2.ipynb google colaboratory notebook appendixiii directory github repository book notice notebook divide step cell section run notebook cell cell process tedious result produce clone openai gpt-2 repository gratifying see run gpt-3 engine line appendix give opportunity code optimize anymore gpt-2 hugging face wrapper encapsulate gpt-2 model useful alternative openai api goal appendix avoid complexity underlie component gpt-2 model explore finally important stress run low level gpt-2 model line obtain result avoid pre packaged version openai gpt-3 api hugging face wrapper get hand dirty understand architecture gpt-2 scratch result deprecation message effort worthwhile industry 4.0 ai guru let begin activate gpu generic text completion gpt-2 step 1 activate gpu activate gpu train gpt-2 345m parameter transformer model activate gpu runtime menu notebook setting vm figure iii.1 gpu hardware accelerator activate gpu prerequisite well performance access world gpt transformer let clone openai gpt-2 repository step 2 clone openai gpt-2 repository openai let download gpt-2 discontinue future maybe access resource point evolution transformer usage move fast foresee market evolve major research lab clone openai github directory vm @title step 2 clone openai gpt-2 repository git clone https://github.com/openai/gpt-2.git cloning repository appear file manager figure iii.2 clone gpt-2 repository click src python file need openai run model figure iii.3 gpt-2 python file run model generic text completion gpt-2 python training file need install train gpt-2 model training gpt-2 language model section

appendix iv custom text completion gpt-2 let install requirement step 3 instal requirement requirement instal automatically @title step 3 instal requirement import os vm restart import os necessary pip3 install -r requirements.txt run cell cell restart vm import os requirement notebook fire 0.1.3 generate command line interface clis regex 2017.4.5 regex usage request 2.21.0 http library tqdm 4.31.1 display progress meter loop ask restart notebook restart let wait check version tensorflow step 4 check version tensorflow gpt-2 345 m transformer model provide openai use tensorflow 1.x lead warning run program ignore run speed thin ice train gpt model modest machine 2020s gpt model reach 175 billion parameter make impossible train efficiently have access supercomputer number parameter continue increase corporate giant research lab facebook ai openai google research brain speed super transformer leave learn un- derstand unfortunately time update model share notebook tensorflow 2.x late tensorflow version old program helpful reason google colaboratory vm preinstalle version tensorflow 1.x tensorflow 2.x tensorflow 1.x notebook @title step 4

check version tensorflow colab tf 1.x tf 2.x instal restart runtime runtime > restart runtime import tensorflow tf output tensorflow 1.x select tf 1.x version display rerun cell sure restart vm rerun cell sure continue time restart vm default version vm tf.2 ready download gpt-2 model step 5 download 345m parameter gpt-2 download train 345m parameter gpt-2 model @title step 5 download 345 m parameter gpt-2 model run code send argument import os runtime restart python3 download_model.py 345 m encounter tensforflow error process ignore warning rerun cell restart vm rerun sure generic text completion gpt-2 path model directory contain information need run model figure iii.4 gpt-2 python file 345m parameter model hparams.json file contain definition gpt-2 model n_vocab 50257 size vocabulary model n_ctx 1024 context size n_embd 1024 embedding size n_head 16 number head n_layer 24 number layer encoder.json vacab.bpe contain tokenize vocabulary bpe word pair neces- sary minute read step 3 train tokenizer subsection chapter 4 pretraine roberta model scratch checkpoint file contain train parameter checkpoint example contain train parameter 1,000

step step 9 train gpt-2 model section appendix iv custom text completion gpt-2 checkpoint file save important file model.ckpt.meta describe graph structure model contain graphdef saverdef retrieve information tf.train.import_meta model.ckpt.index string table key contain tensor value bundleentryproto contain metadata tensor model.ckpt.data contain value variable tensorbundle collection download model intermediate step acti- vate model step 6 7 intermediate instruction section step 6 7 7a intermediate step lead step 8 define activate model want print utf encode text console interact model @title step 6 print utf encode text console want sure src directory @title step 7 project source code import os import runtime restart ready interact gpt-2

model run directly command training gpt-2 language model section appendix iv custom text completion gpt-2 section main aspect code interactive_conditional_samples.py import necessary module require interact model @title step 7a interactive conditional sample src project source code interactive conditional samples /content gpt-2 src interactive_conditional_samples.py file import numpy np import tensorflow tf go intermediate step lead activation model generic text completion gpt-2 step 7b-8 import define model activate interaction model interactive_conditional_samples.py need import module /content gpt-2 src import model sample encoder program model.py define model structure hyperparameter multi attention tf.matmul operation activation function property sample.py process interaction control sample generate make sure token meaningful softmax value blurry like look image low definition sample py contain variable name temperature value sharp increase high probability soften low one sample.py activate k sampling k sampling sort probability distribution predict sequence high probability value head distribution filter k th token tail contain low probability exclude prevent model predict low quality token sample.py activate p sampling language modeling p sampling sort probability distribution instead select word high probability sum subset probability nucleus possible sequence exceed p. encoder.py

encode sample sequence define model encoder.json vocab.bpe contain bpe encoder text decoder open program explore double click interactive_conditional_samples.py function require interact model initialize following information hyperparameter define model model py sample sequence parameter sample.py encode decode sequence interactive_conditional_samples.py restore checkpoint datum define step 5 download 345m parameter gpt-2 model subsection section explore interactive_conditional_samples.py double click experiment parameter model_name model 124 m 345 m rely models_dir models_dir define directory contain model seed set random integer random generator seed set reproduce result nsample number sample return set 0 continue generate sample double click run button cell press ctrl + m. batch_size determine size batch impact memory speed length number token generate text set rely hyperpa- rameter model temperature determine level boltzmann distribution temperature high completion random temperature low result top_k control number token take consideration k step 0 mean restriction 40 recommend value top_p control p. program section scenario parameter explore model_name = 345 m seed = nsample = 1 batch_size = 1 length = 300 temperature = 1 top_k = 0 models_dir = /content gpt-2 model parameter influence model behavior way condition context input generate text completion sequence run notebook default value change code parameter double click program edit save change delete restart vm save program reload wish create interaction scenario program ready prompt interact generic text completion gpt-2 step 9 interact gpt-2 section interact gpt-2 345 m model message system run long google colaboratory main- tain tf 1.x

run model notebook day use gpt-3 engine notebook obsolete use hugging face gpt-2 wrapper example deprecate future meantime gpt-2 use let interact model interact model run interact_model cell @title step 9 interact gpt-2 prompt enter context figure iii.5 context input text completion try type context wish standard gpt-2 model try sentence write emmanuel kant human reason sphere cognition call consider question decline present nature answer transcend faculty mind press enter generate text output relatively random gpt-2 model train

dataset run stochastic model let look line gpt model generate time run grant conception peculiarity second law logic experience measure end apprehension close consciousness solution scholastic perplexity moral religious impossible existence blasphemous stop cell double click run button cell press ctrl + m stop generate text transform code text copy program cell output rich observe fact context enter condition output generate model context demonstration model learn model modify parameter text completion condition context open door transformer model require fine tuning semantic perspective output interesting grammatical perspective output convincing obtain impressive result train model customize dataset appendix iv custom text completion gpt-2 openai gpt-2 github repository https://github.com/openai/gpt-2 n shepperd github repository https://github.com/nshepperd/gpt-2 join book discord space join book discord workspace monthly ask session author appendix iv custom text completion gpt-2 appendix relate chapter 7 rise suprahuman transformers gpt-3 engines describe customize text completion gpt-2 model appendix show build gpt-2 model train interact custom text 12 step open training_openai_gpt_2.ipynb github repository

appendix notice notebook divide 12 step cell appendix run notebook cell cell process tedious result produce clone openai gpt-2 repository gratifying gpt-3 api hugging face wrapper get hand dirty model build train dep- recation message need inside model wrapper api effort worthwhile let begin activate gpu train gpt-2 language model section train gpt-2 model custom dataset encode interact customize model kant.txt dataset chapter 4 pretraine roberta model scratch open notebook run cell cell step 1 prerequisite file refer section available appendixiv directory book github activate gpu google colab notebook runtime menu run google colab explain step 1 activate gpu appendix iii generic text completion gpt-2 custom text completion gpt-2 upload following python file google colaboratory build file manager train.py load_dataset.py encode.py accumulate.py memory_saving_gradients.py file originally come n shepperd github repository https://github.com/ nshepperd gpt-2 download file

appendixiv\gpt-2- train_files directory book github repository n shepperd github repository provide necessary file train gpt-2 model clone n shepperd repository instead clone openai repository add training file need n shepperd repository upload dset.txt google colaboratory build file manager dataset name dset.txt replace content modify program customize input read appendix dataset gpt-2 train_files directory github repository appen- dix kant.txt dataset chapter 4 pretraine roberta model scratch initial step training process step 2 6 initial step training process subsection briefly step 2 6 describe detail ap- pendix iii generic text completion gpt-2 copy dataset model project directory program clone openai gpt-2 repository n shepperd repository @title step 2 clone openai gpt-2 repository git clone https://github.com/nshepperd/gpt-2.git git clone https://github.com/openai/gpt-2.git upload file need train gpt-2 model n shepperd directory program install requirement @title step 3 instal requirement

import os vm restart import os necessary pip3 install -r requirements.txt notebook require toposort topological sort algorithm pip install toposort check tensorflow version sure run version tf 1.x @title step 4 check tensorflow version colab tf 1.x tf 2.x instal restart runtime runtime > restart runtime import tensorflow tf tf 1.x version display rerun cell sure restart vm rerun cell way sure run vm tf 1.x program download 117 m parameter gpt-2 model train dataset @title step 5 download 117 m parameter gpt-2 model run code send argument import os runtime restart python3 download_model.py 117 m create model directory copy dataset 117 m parameter gpt-2 model src directory @title step 6 copy project resource src cp /content dset.txt /content gpt-2 src/ cp -r /content gpt-2 models/ /content gpt-2 src/ goal group resource need train model src project directory n shepperd training file step 7 n shepperd training file training file use come n shepperd github repository upload step 1 prerequisite appendix copy project directory @title step 7 copy n shepperd training files restart notebook instal requirement instead wait check tensorflow version restart vm session restart necessary tedious worthwhile inside code wrapper api custom text completion gpt-2 referfence github

repository https://github.com/nshepperd/gpt-2 import os import runtime restart cp /content train.py /content gpt-2 src/ cp /content load_dataset.py /content gpt-2 src/ cp /content encode.py /content gpt-2 src/ cp /content accumulate.py /content gpt-2 src/ cp /content memory_saving_gradients.py /content gpt-2 src/ training file ready activate let explore start encode.py step 8 encode dataset dataset encode train double click encode.py display file google colaboratory encode.py load dset.txt call load_dataset function load_dataset.py load_dataset import load_dataset chunk = load_dataset(enc args.in_text args.combine encoding = arg encode.py load openai encoding program encode.py encode dataset enc = encoder.get_encoder(args.model_name models_dir encode dataset save numpy array store out.npz

npz numpy zip archive array generate encoder import numpy np dataset load encode save out.npz run cell @title step 8 encode dataset import os import runtime restart python /content gpt-2 src encode.py dset.txt out.npz gpt-2 117 m model ready train step 9 train gpt-2 model train gpt-2 117 m model dataset send encode dataset program @title step 9 train model model save 1000 step import os import runtime restart python train.py --dataset out.npz run cell train stop train model save 1,000 step training exceed 1,000 step stop save model checkpoint /content/ gpt-2 src checkpoint run1 check list file step 10a copy training files cell notebook stop training double click run button cell training end train parameter save stop train model 1,000 step ctrl + m. program stop save train parameter convert code text copy code cell display following message figure iv.1 save train gpt-2 model automatically program manage optimizer gradient /content gpt-2 src memory_saving gradients.py /content gpt-2 src accumulate.py program train.py

contain complete list parameter tweak modify training pro- cess run notebook change wish experiment training parameter obtain well result gpt-3 model generate sample read training point gpt-2 training run system generate sample find enlightening world thing representation

world custom text completion gpt-2 representation world human create ai learn interesting let continue experiment create directory training model step 10 create training model directory section create temporary directory model store information need rename replace directory gpt-2 117 m model download start create temporary directory name tgmodel @title step 10 create training model directory create training model directory name tgmodel run_dir = /content gpt-2 models tgmodel os.path.exists(run_dir copy checkpoint file contain train parameter save train model step 9 train model subsection section @title step 10a copy training file cp /content gpt-2 src checkpoint run1 model-1000.data-00000 of-00001 cp /content gpt-2 src checkpoint run1 checkpoint /content gpt-2 models/ cp /content gpt-2 src checkpoint run1 model-1000.index /content gpt-2/ cp /content gpt-2 src checkpoint run1 model-1000.meta /content gpt-2/ tgmodel directory contain train parameter gpt-2 model describe file content step 5 download 345 m parameter gpt-2 model ap- pendix iii generic text completion gpt-2 retrieve hyperparameter vocabulary file gpt-2 117 m model @title step 10b copy openai gpt-2 117 m model file cp /content gpt-2 models/117m encoder.json /content gpt-2 model tgmodel cp /content gpt-2 models/117m hparams.json /content gpt-2 model tgmodel cp /content gpt-2 models/117 m vocab.bpe /content gpt-2 model tgmodel tgmodel directory

contain complete customize gpt-2 117 m model step rename original gpt-2 model download set model 117 m @title step 10c rename model directory mv /content gpt-2 models/117 m /content gpt-2 models/117m_openai mv /content gpt-2 model tgmodel /content gpt-2 models/117 m train model clone openai gpt-2 repository run let interact model step 11 generate unconditional sample section interact gpt-2 117 m model train dataset generate unconditional sample require input enter context paragraph obtain conditional text completion response train model let run unconditional sample @title step 11 generate unconditional sample import os import runtime restart python generate_unconditional_samples.py --model_name 117 m prompt enter context sentence

unconditional sample gen- stop cell double click run button cell press ctrl + m. result random make sense grammatical perspective semantic point view result interesting provide context process remarkable invent post write title date invent organization address produce topic imagine web link line incredible join july 17th 2013 custom text completion gpt-2 offtopic link essential research research support paper peer review research back claim design research unjustifiably accommodate scientific uncertainty research persuade open door science participation science join 11th 2014 location montana areajoined 11th 2014posts 1729location montana post fri dec 26 2017 9:18 pm post subject click i. synopsis establish review group new research paradigm preferred alternative brepg group lead author john obi australian usa chartered institute tropical climate change research marco xiao china department sociology/ ajax international institute tropical climate change research shanghai university jackie gu pacific university interselicitas de nasa frozen planet research research center oak ridge national laboratory

dr. obi state conclusion indicate existence new peer review asan rubie study predispose journal publisher read scientific publisher constantly seek consignment affiliate large target certain person current practice supply book peer review commonly overrule specialist spread sufficient spread impact peer review expert level organization increase extent exposure term deviation source profile finding require result unconditional random text generator interesting convincing step 12 interactive context completion example run conditional sample context enter condition model think want complete text generate tailor paragraph run cell explore magic @title step 12 interactive context completion example import os import runtime restart python interactive_conditional_samples.py --temperature 0.8 --top_k 40 necessary minute parameter step 9 interact gpt-2 appendix iii generic text completion gpt-2 program prompt enter context figure iv.2 context input text completion let enter paragraph write emmanuel kant step 9 interact gpt-2 appendix iii generic text completion gpt-2 human reason sphere cognition call consider question decline present nature answer transcend faculty mind

press enter generate text previously output change run structure logical make transformer attractive time result random impressive let look line gpt-2 model produce nature reason kind power mind call think understand reason thing apprehend seek understand object mind apprehend apprehend thing speak reason understanding understanding human reason knowledge understanding natural faculty understand human reason natural faculty reason understanding human reason see know thing natural sense natural sense understand source knowledge understand reason thing understand understand thing comprehend understand mean reason thing understand know thing understand stop cell double click run button cell enter ctrl + m. custom text completion gpt-2 wow doubt anybody difference text completion produce train gpt-2 model human generate different output run output example question oblige respond true mind ignorant truth proposition ought accept proposition give question answer impossible man believe existence mind receive mind man far judge subject philosophical inquiry mind

mind receive mind man far judge subject philosophical inquiry mind mind man opinion consider important subject understand understand think model outperform human abstract exercise philosophy reason limit text vary run excellent fit single need everyday life draw conclusion experiment train transformer model produce text completion human level gpt-2 model reach human level text generation complex abstract text context efficient way condition model demonstrate expect text completion text generation base text conditioning context sentence text human level mean fit need locally interesting globally effective point try enter conditioning text context example experiment text completion train model datum replace content dset.txt file happen bear mind train gpt-2 model react like human enter short incomplete uninteresting tricky context obtain puzzled bad result gpt-2 expect good real life openai gpt-2 github repository https://github.com/openai/gpt-2 n shepperd github repository https://github.com/nshepperd/gpt-2

join book discord space join book discord workspace monthly ask session author appendix v answer chapter 1 transformer industrial revolution true false false eras history overlap industrial revolution focus make world digital fourth industrial revolution begin connect system machine bot robot algorithm fourth industrial revolution connect true false true lead increase automate decision require industry 4.0 developer ai development true false true project ai online service require development industry 4.0 developer implement transformer scratch true false true project standard online service api satisfy need project satisfactory solution project case instead developer customize model significantly work scratch necessary learn transformer ecosystem hugging face example true false false corporation policy work google cloud ai microsoft azure ai hugging face tool company know ad- vance case decide 6 ready use transformer api satisfy need true false true effective false transformer model train answer question company accept transformer ecosystem developer know well true false false company accept developer suggest safe cover basis possible 8 cloud transformer mainstream true false 9 transformer project run laptop true false true prototype example false project involve thousand user 10 industry 4.0 ai specialist flexible true false chapter 2 getting start architecture transformer model nlp transduction encode decode text representation true false true nlp transduction convert sequence written oral numerical rep- resentation process decode result text natural language understanding nlu subset natural language processing language modeling algorithm generate probable sequence word base input transformer customize lstm cnn layer true false false transformer contain lstm cnn 5 transformer contain lstm cnn layer true false 6 attention examine token sequence true false transformer use positional encoding true false false transformer use positional encoding 8 transformer contain feedforward network true false mask multi headed attention component decoder transformer prevent algorithm parse give position see rest sequence 10 transformer analyze long distance dependency

position see rest sequence 10 transformer analyze long distance dependency well lstms true

false chapter 3 fine tune bert models bert stand bidirectional encoder representation transformers true false bert step framework step 1 pretraine step 2 fine tuning true false fine tune bert model imply training parameter scratch true false false bert fine tuning initialize train parameter pretraine bert pretrain downstream task true false bert pretrain masked language modeling mlm true false bert pretrain sentence prediction nsp true false bert pretrain mathematical function true false answer question 8 question answer task downstream task true false 9 bert pretraine model require tokenization true false 10 fine tune bert model take time pretraine true false chapter 4 pretraine roberta model roberta use byte level byte pair encoding tokenizer true false 2 train hugging face tokenizer produce merges.txt vocab.json true false roberta use token type id true false distilbert 6 layer 12 head true false 5

transformer model 80 million parameter enormous true false false 80 million parameter small model train tokenizer true false false tokenizer train bert like model decoder layer true false false bert contain encoder layer decoder layer 8 mlm predict word contain mask token sentence true false 9 bert like model self attention sublayer true false false bert self attention layer 10 data collator helpful backpropagation true false chapter 5 downstream nlp task transformers machine intelligence use datum human prediction true false true false true case machine intelligence surpass human process massive amount datum extract meaning perform range task centu- rie human process false nlu human access information sense machine intelligence rely human provide type medium superglue difficult glue nlp model true false boolq expect binary answer true false wic stand words context true false recognize textual entailment rte detect sequence entail 6 winograd schema predict verb spell correctly true false false winograd schema mainly apply pronoun disambiguation transformer model occupy rank glue superglue true false answer question 8 human baseline standards define tough attain superglue true false transformer model beat superglue human baseline standard true false true false false transformer model beat human baseline glue su- perglue future true

set high benchmark standard progress field nlu 10 variant transformer model outperform rnn cnn model true false true know happen future ai chapter 6 machine translation transformer machine translation exceed human baseline true false false machine translation challenging nlp ml task machine translation require large dataset true false need compare transformer model dataset true false false way compare different model use dataset bleu french word blue acronym nlp metric true false true bleu stand bilingual evaluation understudy score make easy re- smooth technique enhance bert true false german english english german machine translation true false false represent german translate language process represent

translation true false false represent german translate language process represent english translate language language structure original transformer multi head attention sublayer head true false false attention sublayer head 8 original transformer encoder layer true false original transformer encoder layer decoder layer true false false decoder layer 10 train transformer decoder true false true architecture bert contain encoder chapter 7 rise suprahuman transformers gpt-3 engines zero shot method train parameter true false false parameter train gradient update perform run zero shot model true false gpt model decoder stack true false impossible train 117 m gpt model local machine true false false train chapter impossible train gpt-2 model specific dataset true false false train chapter 6 gpt-2 model condition generate text true false false implement chapter gpt-2 model analyze context input produce completion content true/ answer question 8 interact 345 m gtp parameter model machine few gpu true false false interact model size chapter supercomputer 285,000 cpu exist true false 10 supercomputer thousand gpu game changer ai true false true

thank able build model increase number param- eter connection chapter 8 apply transformer legal financial document ai text summarization t5 model encoder stack like bert model true false t5 model encoder decoder stack true false t5 model use relative positional encoding absolute positional encoding true false text text model design summarization true false text

text model apply prefix input sequence determine nlp task t5 model require specific hyperparameter task true false advantage text text model use hyperparameter nlp task true false 8 t5 transformer contain feedforward network true false hugging face framework make transformer easy implement true false 10 openai transformer engine game changer true false true openai produce wide range ready use engine codex language code davinci general purpose engine chapter 9 matching tokenizer dataset tokenize dictionary contain word exist language true false pretrained tokenizer encode dataset true false good practice check database true false good practice eliminate obscene datum dataset true false good practice delete datum contain discriminate assertion true false raw dataset produce relationship noisy content useful standard pretraine tokenizer contain english vocabulary past 700 year answer question 8 old english create problem encode datum tokenizer train modern medical type jargon create problem encode datum to- kenizer train modern english true false 10 control output encode datum produce pretrained tokenizer good chapter 10 semantic role labeling bert- semantic role labeling srl text generation task true false 2 predicate noun true false verb predicate true false 5 modifier adverb true false 6 modifier location true false bert base model contain encoder decoder stack true false 8

bert base srl model standard input format true false transformer solve srl task true false chapter 11 let data talking story question answer train transformer model answer question true false question answering require research perfect true false name entity recognition ner provide useful information look mean- ingful question true false semantic role labeling srl useless prepare question true false 5 question generator excellent way produce question true false implement question answering require careful project management true false electra model architecture gpt-2 true false 8 electra model architecture bert train discriminator answer question ner recognize location label loc true false 10

ner recognize person label person true false chapter 12 detect customer emotion necessary

pretrain transformer sentiment analysis true false 2 sentence positive negative neutral true false principle compositionality signify transformer grasp sentence understand true false roberta large design improve pretraine process transformer model 5 transformer provide feedback inform customer satisfied sentiment analysis product service consistently negative help appropriate decision improve offer true false model fail provide good result task require training fine tuning change model true false chapter 13 analyze fake news transformers news label fake news fake true false news everybody agree accurate true false transformer run sentiment analysis tweet true false key entity extract facebook message distilbert model run key verb identify youtube chat bert base model run srl emotional reaction natural response fake news true false rational approach fake news help clarify position true false 8 connect transformer reliable website help somebody understand news fake true false transformer summary reliable website help understand topic label fake news true false 10 change world use ai good true false answer question chapter 14 interpret black box transformer bertviz show output layer bert model true false false bertviz display output layer bertviz show attention head layer bert model true false bertviz show token relate true false lit show inner working attention head like bertviz true false false lit make non probing prediction probing way algorithm predict language representation true false ner probe task true false pca umap non probing task true false 8 lime model agnostic true false transformer deepen relationship token layer layer true false 10 visual transformer model interpretation add new dimension interpretable ai true/ chapter 15 nlp task agnostic transformer reformer transformer model contain encoder true false false reformer transformer model contain encoder reformer transformer model contain decoder true false false reformer transformer model contain encoder decoder

decoder true false false reformer transformer model contain encoder decoder input store layer layer reformer model true false false input recompute level save memory deberta transformer model disentangle content position true false necessary test hundred

pretrained transformer model choose project true false true false try model choose reliable model implement fit need late transformer model good true false true false lot research produce transformer experimental model short live late model exceed performance precede model well transformer model nlp task multi task transformer true false personal decision risk assessment critical aspect project 8 transformer model need fine tune true false false gpt-3 engine zero shot model answer question openai gpt-3 engine perform wide range nlp task fine tuning true/ 10 well implement ai algorithm local server true false false depend project risk assessment chapter 16 emergence transformer drive ai copilot generate code automatically exist true false false github copilot example production 2 ai copilot replace human true false true false ai task sale support maintenance domain complex task require human intervention gpt-3 engine task true false false gpt-3 engine wide variety task transformer train recommender true false true transformer go language sequence sequence domain transformer process language true false false transformer train language sequence analyze type sequence 6 transformer sequence contain word true false false language sequence process transformer work number vision transformer equal cnn true false false transformer deep neural network equal cnn computer vision 8 ai robot computer vision exist true false false example robot computer vision begin surface military ap- impossible produce python source code automatically true false false microsoft openai join produce copilot write python code 10 day copilot robot true false true false remain challenge human bot robot grow ai ecosystem join book discord space join book discord workspace monthly ask session author subscribe online digital library access 7,000 book video industry lead tool help plan personal development advance career information visit website spend time learn time code practical ebook videos 4,000 industry professional improve learning skill plans build especially free

4,000 industry professional improve learning skill plans build especially free ebook video month fully searchable easy access vital information copy paste print bookmark content

www.packt.com read collection free technical article sign range free newsletter receive exclusive discount offer packt book ebook enjoy enjoy book interested book packt machine learning pytorch scikit learn yuxi hayden liu explore framework model technique machine learn datum use scikit learn machine learning pytorch deep learning train machine learning classifier image text build train neural network transformer boost algorithm discover good practice evaluate tune model predict continuous target outcome regression analysis dig deeply textual social medium datum sentiment analysis book enjoy machine learn time series python understand main class time series learn detect outlier pattern choose right method solve time series problem characterize seasonal correlation pattern autocorrelation statistical grip time series datum visualization understand classical time series model like arma arima implement deep learning model like gaussian process transformer state of- art machine learning model familiar library like prophet xgboost tensorflow books enjoy deep reinforcement learning hands second edition understand deep learning context rl implement complex deep learning model evaluate rl method include cross entropy dqn actor critic trpo ppo ddpg d4pg build practical hardware robot train rl method $ 100 discover microsoft textworld environment interactive fiction game plat- use discrete optimization rl solve rubik cube teach agent play connect 4 alphago zero explore late deep rl research topic include ai chatbot discover advanced exploration technique include noisy network network dis- book enjoy share thought finish transformers natural language processing second edition love hear thought purchase book amazon click straight amazon review page book share feedback leave review site purchase review important tech community help sure deliv- ere excellent quality content packt search author like interested author packt visit authors.packtpub.com apply today work thousand developer tech

author packt visit authors.packtpub.com apply today work thousand developer tech professional like help share insight global tech community general appli- cation apply specific hot topic recruit author submit idea 345m parameter gpt-2 model downloading 475 476

zero shot zs model 17 fine tuning ft model 173 175