Precog Task For NLP

Vikash Kumar Prasad

GTM Buddy,
vikashp@gtmbuddy.ai

Abstract. We choose the task of NLP to find the similarity scores. We first show the work of Word similarity scores then phrase and sentence similarity and lastly the bonus task is performed. For each of the sub part task we summarize the findings, methodology used, and show the results.

Keywords: Fasttext, BERT, Natural Language Processing, Word Embeddings, Semantic Similarity, Ollama

1 Word Similarity Scores

For part 1 we use the approach of embeddings to derive the similarity scores and for part 2 we use BERT to derive the similarity scores. For each part we detail the methodology as in part1 and part 2. We then combine our findings and results.

1.1 Methodology

We used monolingual dataset of wikitext [6] to create embeddings using the gensim library. We consider 1 million tokens to create these embeddings. We create FastText [1], and Word2Vec [7] embeddings and compare the results with the SimLex [4] dataset which are human annotated. For all the embeddings we keep the following parameters same.

1. Vector Size of 100
2. Window Size of 5
3. Minimum Count of 5
4. SG=0(as we want to capture more semantic similarities and less of relatedness).
5. 30 epochs

We utilize cosine similarity to measure the similarity between two words represented as vectors, focusing on the angle between them rather than their magnitudes. This makes it particularly effective for assessing semantic similarity in high-dimensional spaces, such as the word embeddings we created.
Additionally, we employ an ontology-based method using WordNet, which offers structured knowledge through its synsets and hierarchical relationships. By

leveraging the Wu-Palmer similarity measure, we quantify the semantic similarity of word pairs based on their Least Common Subsumer (LCS) within the ontology, providing meaningful similarity scores.
In SimLex, we assign a target of 1 to scores above 5 and 0 to scores less than or equal to 5. For similarity scores, we give a value of 1 for scores greater than 0.5 and 0 for scores less than or equal to 0.5. We then use accuracy and F1 score metrics to identify the best method, whether FastText, Word2Vec, or the ontology-based approach using WordNet.
For part b we use GLoVE[9] embeddings to derive vectors for semantic similarity testing. We use cosine similarity to test SimLex dataset.

1.2 Findings

In measuring semantic similarity, Word2Vec and FastText perform relatively poorly Table1, with accuracy and F1 scores of around 0.53-0.54 and 0.28-0.29, respectively. This is due to their reliance on local context windows, which limits their ability to capture nuanced semantic relationships. In contrast, ontology-based approaches like WordNet leverage structured, human-curated relationships, resulting in a noticeable improvement (accuracy 0.56, F1 score 0.54) by providing a more comprehensive understanding of word meanings. GloVe, however, achieves the highest scores (accuracy 0.61, F1 score 0.60) by training on global word co-occurrence, allowing it to capture both syntactic and semantic relationships more effectively. This global approach enables GloVe to understand complex word associations, making it more suitable for semantic similarity tasks than models trained solely on local context.

Method	**Accuracy**	**F1 Score**
FastText	0.54	0.28
Word2Vec	0.53	0.29
WordNet	0.56	0.54
GLoVE	0.61	0.60

Table 1: Performance Metrics for Different Methods

2 Phrase Similarity and Sentence Similarity

For phrase similarity and sentence similarity, i.e to classify whether two phrases are similar or not we treat it as a classification problem and we first try with classical machine learning problems and then use cross encoder model approach. We first explain an age old problem of NLI and how cross encoder models work.Cross-encoder models are highly effective in Natural Language Inference (NLI), sentence similarity, and determining whether a summary is contained within a larger text due to their ability to encode sentence pairs jointly. In

NLI, cross-encoders improve classification accuracy by capturing nuanced interactions between premise-hypothesis pairs, allowing for a deeper understanding of their semantic relationship [13]. For sentence similarity tasks, this joint encoding approach enables more precise similarity scoring, as seen in applications like paraphrase detection [10].Furthermore, in tasks like summarization, cross-encoders can verify if essential information from a summary is represented within the original text, basically to detect hallucinations, making them useful for quality control in summarization and retrieval contexts [8]. These capabilities demonstrate the advantage of cross-encoders over independently encoded sentence representations across multiple NLP tasks.

2.1 Methodology

2.2 Approach

We apply the following approach for both phrase and sentence similarity tasks for classical machine learning models:

1. Extract embeddings for each word in the phrase or sentence using the pre-trained GloVe model.
2. Compute the average embedding for each phrase and sentence.
3. Concatenate the phrase and sentence embeddings for phrase and sentence similarity.
4. Train classical machine learning models on the concatenated embeddings for a binary classification task.

We notice a poor performance when we test the classical models for phrase similarity Fig 1 across various metrics and same wise for sentence similarity Fig 2.

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC
dummy	Dummy Classifier	0.4998	0.5000	0.1000	0.0499	0.0666	0.0000	0.0000
svm	SVM - Linear Kernel	0.4839	0.4660	0.6110	0.4692	0.4795	-0.0325	-0.0427
nb	Naive Bayes	0.4757	0.4633	0.4818	0.4758	0.4784	-0.0485	-0.0486
lr	Logistic Regression	0.4661	0.4546	0.4664	0.4661	0.4661	-0.0677	-0.0677
ridge	Ridge Classifier	0.4659	0.4547	0.4655	0.4659	0.4655	-0.0681	-0.0682
lda	Linear Discriminant Analysis	0.4657	0.4547	0.4651	0.4656	0.4652	-0.0685	-0.0686
ada	Ada Boost Classifier	0.4417	0.4114	0.4451	0.4413	0.4427	-0.1167	-0.1170
knn	K Neighbors Classifier	0.4045	0.3829	0.4092	0.4054	0.4072	-0.1910	-0.1912
gbc	Gradient Boosting Classifier	0.3452	0.2883	0.3598	0.3494	0.3544	-0.3097	-0.3101
qda	Quadratic Discriminant Analysis	0.3158	0.2456	0.3256	0.3192	0.3223	-0.3684	-0.3686
et	Extra Trees Classifier	0.3039	0.1763	0.2999	0.3019	0.3008	-0.3921	-0.3923
lightgbm	Light Gradient Boosting Machine	0.2948	0.1843	0.3052	0.2988	0.3018	-0.4104	-0.4108
xgboost	Extreme Gradient Boosting	0.2929	0.1771	0.2966	0.2942	0.2952	-0.4141	-0.4145
dt	Decision Tree Classifier	0.2927	0.2600	0.2215	0.2578	0.2382	-0.4145	-0.4191
rf	Random Forest Classifier	0.2889	0.1709	0.2966	0.2922	0.2943	-0.4223	-0.4225

Fig. 1: Classical Models Performance on Phrase Similarity Detection

	Model	Accuracy	AUC	Recall	Prec.	F1	Kappa	MCC	TT (Sec)
gbc	Gradient Boosting Classifier	0.5749	0.5302	0.1414	0.5776	0.2271	0.0645	0.0951	69.2230
lr	Logistic Regression	0.5671	0.5423	0.1926	0.5281	0.2821	0.0603	0.0760	1.0240
ridge	Ridge Classifier	0.5660	0.5408	0.1961	0.5241	0.2853	0.0588	0.0736	0.1330
lda	Linear Discriminant Analysis	0.5652	0.5403	0.1995	0.5209	0.2884	0.0579	0.0718	0.2730
ada	Ada Boost Classifier	0.5622	0.5292	0.2185	0.5114	0.3059	0.0562	0.0673	12.5730
nb	Naive Bayes	0.5424	0.5445	0.4384	0.4805	0.4584	0.0637	0.0639	0.1380
svm	SVM - Linear Kernel	0.5413	0.5363	0.2893	0.5394	0.2746	0.0312	0.0488	0.5610
qda	Quadratic Discriminant Analysis	0.5181	0.5066	0.4287	0.4523	0.4400	0.0176	0.0177	0.3490
knn	K Neighbors Classifier	0.5038	0.4636	0.2951	0.4137	0.3444	-0.0369	-0.0383	1.2720
xgboost	Extreme Gradient Boosting	0.4862	0.4403	0.3074	0.3955	0.3459	-0.0663	-0.0677	3.5100
rf	Random Forest Classifier	0.4602	0.4042	0.3107	0.3685	0.3371	-0.1126	-0.1138	16.0250
dt	Decision Tree Classifier	0.4545	0.4321	0.3500	0.3745	0.3618	-0.1135	-0.1137	4.3490
et	Extra Trees Classifier	0.4527	0.3848	0.3088	0.3606	0.3327	-0.1265	-0.1276	5.7150

Fig. 2: Classical Models Performance on Sentence Similarity Detection

We take a different approach to improve accuracy by using cross-encoder models. For both phrase and sentence similarity tasks, we apply the following method:

1. We utilize four popular transformer-based models: BERT [2], RoBERTa [5], DistilBERT [12], and DistilRoBERTa [12]. Each model is initially trained for 10 epochs on the phrase similarity task and 5 epochs on the sentence similarity task, given the large dataset size.
2. We then select the best model based on validation accuracy.
3. We train the selected model for longer epochs 30 epochs for phrase similarity and 15 epochs for sentence similarity.

2.3 Findings

We observe the below:

1. For phrase similarity Fig3 we observe that Roberta Base performs the best at 0.73 accuracy.
2. We observed the same for sentence similarity Fig4, observing a tectonic shift in accuracy measures for sentence similarity.
3. For phrase similarity we find the best accuracy is reached at 0.75 using Roberta Base Fig5.
4. For sentence similarity we find the best accuracy is reached at 0.95 using Roberta Base Fig6.
5. Off all the approaches(classical machine learning models) cross encoder models perform the best.

3 Bonus Task

3.1 Using Transformers

We have already experimented with Transformers experimenting with 4 different models to get the best model.

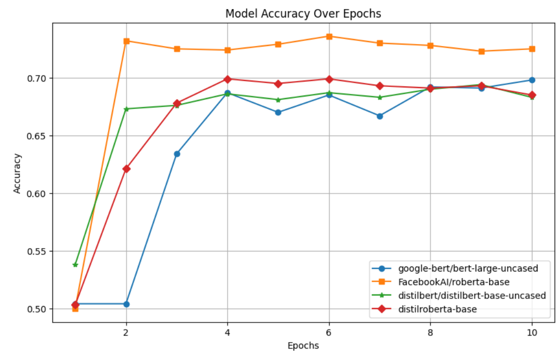


Fig. 3: Models Performance Across 10 Epochs For Phrase Similarity

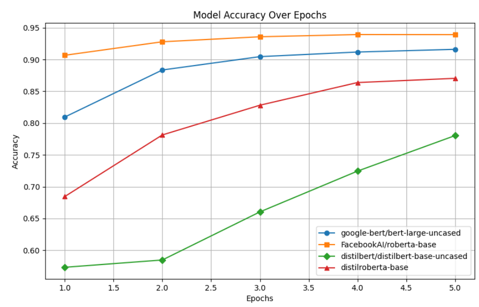


Fig. 4: Models Performance Across 5 Epochs For Sentence Similarity

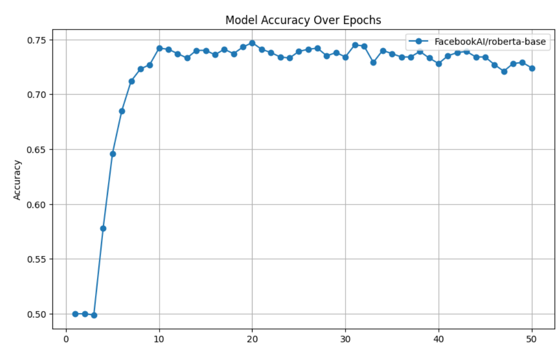


Fig. 5: Roberta Base Performance Across 50 Epochs For Phrase Similarity

3.2 Using LLMs

Before prompting we try and obtain embeddings from two LLMs gemma:2B[11] and llama3.2:1B[3] for both the phrase similarity and sentence similarity on their

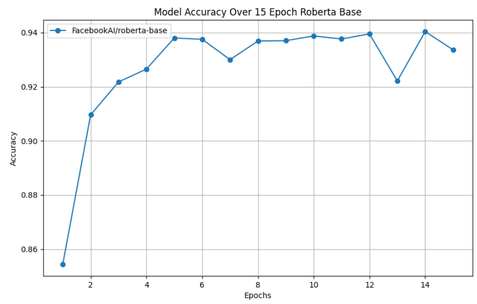


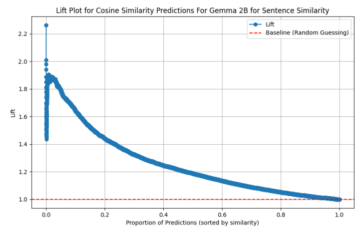
Fig. 6: Roberta Base Performance Across 15 Epochs For Sentence Similarity

training split part. We wanted to observe that on an unsupervised manner how well do the embeddings from instruction based LLMs are able to separate from similar and non-similar phrases and sentences.
We did it in the following way:

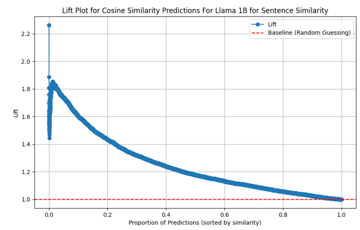
1. We use open source server Ollama to deploy LLAMA3.2:1B and Gemma:2B on a Tesla T4 GPU linux environment.
2. Batchwise we extract the embeddings, as there were 7000+phrases and 49000+ sentences to process. A glimpse of how fast the processing was done Fig 7.

Fig. 7: Batchwise Extraction Of Embeddings For Phrase And Sentence Similarity

3. Once done we extracted the cosine similarity and plotted the lift curve to explain the effect of embeddings from these models.
4. On comparison of Gemma:2B embeddings with LLama3.2:1B for sentence similarity we find them identical. We observe that both the models perform well in distinguishing dissimilar sentences for higher cosine scores but don't perform well for lower cosine scores. Meaning the lower cosine score cannot be trusted for these models Fig 8.
5. On the other hand for phrase similarity the performance is very poor implying it's really difficult to distinguish phrases for the embeddings from these models, mostly due to the fact that phrases are unformed sentences or part of sentences. On closer analysis we notice that for both higher and lower cosine similarities the embeddings are not able to distinguish between similar and dissimilar phrases Fig9.

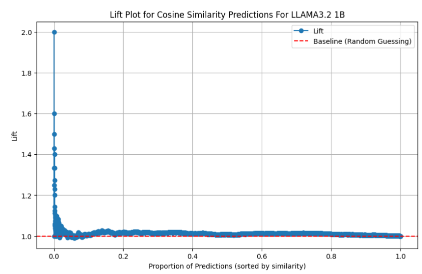


(a) Gemma2B Model For Sentence Similarity

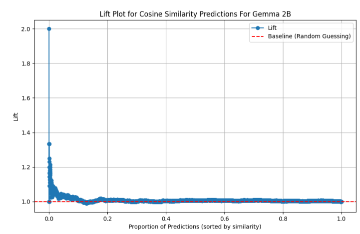


(b) Llama Model For Sentence Similarity

Fig. 8: LIFT Analysis For LLMs Embeddings For Sentence Similarity



(a) LLama Model For Phrase Similarity



(b) Gemma2B Model For Phrase Similarity

Fig. 9: LIFT Analysis For LLMs Embeddings For Phrase Similarity

3.3 Prompting

We also experiment with prompting; however, since prompt-based responses are slow, we limit this approach to 1,000 samples to assess its impact for sentence similarity. We use the below detailed prompt:

f””” On a scale of 0 to 1, how similar are these sentences semantically and return just the score in float? Sentence 1: sent1 Sentence 2: sent2

Think step by step: 1. Compare the main topics/subjects 2. Compare the actions/verbs 3. Compare the context and meaning 4. Consider synonyms and related concepts

Provide a final similarity score between 0 and 1, where: 0 = completely different meaning 1 = identical meaning ”””

”sent1” and ”sent2” represents the sentences we are trying to extract similarity. We then parse through the text to obtain the score. We obtain the lift plot Fig10 to observe that the Llama 1B model performs well for a small subset of highly similar sentence pairs, with a strong initial lift. However, as more predictions are included, the lift drops to nearly the random baseline, indicating that the model struggles to maintain accuracy across most sentence pairs and performs close to random guessing for the majority.

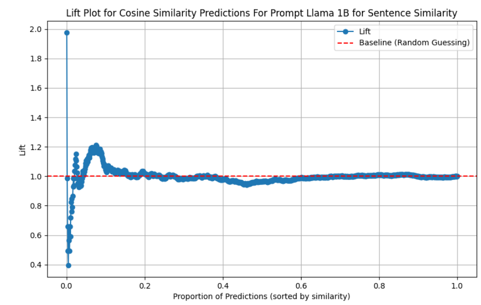


Fig. 10: LIFT Plot For Sentence Similarity Using Prompt For LLAMA3.2:1b Model

3.4 Overall Findings

Classical machine learning approaches struggle to achieve satisfactory metrics, resulting in low accuracy and F1 scores for both phrase and sentence similarity tasks. In contrast, transformer models using a cross-encoder architecture, specifically with a RoBERTa Base backend, perform significantly better. Embeddings from large language models show moderate performance: they are adequate for sentence similarity but only provide baseline results for phrase similarity, as seen in the lift plots Table2.

Approach	**F1**	**Accuracy**
Classical ML Sentence Sim	0.22	0.57
Classical ML Phrase Sim	0.57	0.49
Cross Encoder Phrase Sim	0.73	0.74
Cross Encoder Sentence Sim	0.9275	0.9222

Table 2: Comparison of Different Approaches On Test Data

References

1. Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. 5:135–146, 2017.
2. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. pages 4171–4186, 2019.
3. Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh

Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Cantón Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab A. AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriele Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guanglong Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Laurens Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Ju-Qing Jia, Kalyan Vasuden Alwala, K. Upasani, Kate Plawiak, Keqian Li, Ken-591 neth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline C. Muzzi, Mahesh Babu Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melissa Hall Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri S. Chatterji, Olivier Duchenne, Onur cCelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasić, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Chandra Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yiqian Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zhengxu Yan, Zhengxing Chen, Zoe Papakipos, Aaditya K. Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adi Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Ben Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly

Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Shang-Wen Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzm'an, Frank J. Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory G. Sizov, Guangyi Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Han Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kaixing(Kai) Wu, U KamHou, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, A Lavender, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollár, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sung-Bae Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Andrei Poenaru, Vlad T. Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xia Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The llama 3 herd of models. *ArXiv*, abs/2407.21783, 2024.

4. Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. 41(4):665–695, 2015.
5. Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
6. Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.
7. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. 2013.
8. Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.
9. Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. pages 1532–1543, 2014.
10. Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.
11. Gemma Team Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, L'eonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ram'e, Johan Ferret, Peter Liu, Pouya Dehghani Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stańczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Boxi Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Christoper A. Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozi'nska, D. Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Pluci'nska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost R. van Amersfoort, Josh Gordon, Josh Lipschultz, Joshua Newlan, Junsong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, L. Sifre, L. Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Gorner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, S. Mc Carthy, Sarah Perrin, S'ebastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomás Kociský, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei,

Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Brian Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeffrey Dean, Demis Hassabis, Koray Kavukcuoglu, Cl'ement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size. *ArXiv*, abs/2408.00118, 2024.

12. Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. 2019.
13. Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. 2018.