Pixie: Preference in Implicit and Explicit Comparisons

Amanul Haque and **Vaibhav Garg** and **Hui Guo** and **Munindar P. Singh**
Department of Computer Science
North Carolina State University
Raleigh, NC 27695, USA
{ahaque2, vgarg3, hguo5, mpsingh}@ncsu.edu

Abstract

We present Pixie, a manually annotated dataset for preference classification comprising 8,890 sentences drawn from app reviews. Unlike previous studies on preference classification, Pixie contains *implicit* (omitting an entity being compared) and *indirect* (lacking comparative linguistic cues) comparisons. We find that transformer-based pretrained models, finetuned on Pixie, achieve a weighted average F1 score of 83.34% and outperform the existing state-of-the-art preference classification model (73.99%).

1 Introduction

Online user reviews contain a cornucopia of information on user expectations about a product. Users often express their opinions on a product by comparing it against competitors. Understanding preferences in natural language is crucial in capturing user's opinions and expectations. Previous studies show that app reviews include rich insights about user expectations and problems of mobile apps that are valuable for app developers (Palomba et al., 2015; Maalej and Nabil, 2015; Guo and Singh, 2020). We found that app reviews often include comparative sentences, from which we can determine a reviewer's preferences.

Identifying the preferred entity from an app review involves (1) *Comparative Sentence Identification* (CSI) (Jindal and Liu, 2006), i.e., identifying sentences that contain a comparison, and (2) *Comparative Preference Classification* (CPC) (Ganapathibhotla and Liu, 2008; Panchenko et al., 2019), i.e., identifying the preferred entity in a comparative sentence. We focus on the second task.

Prior work on CPC focuses on *explicit* comparisons, where all compared entities are explicitly mentioned. Extracting comparative sentences by matching keywords or patterns (Jindal and Liu, 2006; Li et al., 2017; Feldman et al., 2007) overlooks *indirect* comparisons which lack comparative quantifiers and adjectives.

	Sentence	App
S_1	Bye ***Uber***, hello ***Lyft***.	*Uber*
S_2	Does ***this app*** really need to be 260 MB when the ***Marriott app*** is only 47 MB?	*Hilton Honors*
S_3	Beats the pants off ***pandora***.	*Spotify*
S_4	I think that ***it's*** a lot more fun than ***temple Run***.	*Subway Surfers*

Table 1: Example comparative sentences from reviews.

Staab and Hahn (1997) identify *omitted complement* as a comparative sentence type that has been overlooked by prior research. An omitted complement refers to one of the entities under comparison that is omitted but can be inferred based on the context. We have found that comparative sentences in user-generated text such as reviews sometimes imply instead of explicitly mentioning the target entity being reviewed (e.g., S_3 in Table 1). Comparisons in reviews often lack comparative linguistic cues, such as comparative quantifiers, adjectives, or structures (i.e., indirect, e.g., S_1 in Table 1). Such sentences are comparative by virtue of expressing a preference and are common in reviews but have been understudied by prior research.

We present Pixie (Preference in Implicit and Explicit Comparisons), a dataset for preference classification, created from online user reviews. As shown in Table 1, Pixie includes indirect comparisons (i.e., sentences lacking comparative linguistic cues, e.g., S_1) and *implicit* comparisons (omitting compliments, i.e., mentioning only one entity being compared, e.g., S_3) in addition to *direct* comparisons (comparing entities with a direct comparative structure, e.g., S_4) and *explicit* comparisons (mentioning both entities being compared, e.g., S_2).

We experiment with traditional machine learning methods and transformer-based models on Pixie. We use segment embeddings to demarcate the compared entities before fine-tuning the transformer-based models. We also compare our results with ED-GAT (Ma et al., 2020), a state-of-the-art model for preference classification. We find that transformer-based pretrained language models, fine-tuned on Pixie, achieve a higher F1-score (83.34%) than the state-of-the-art (F1-score 73.99%) or traditional machine learning models (F1-score 71.86%) trained on Pixie. Further error analysis of misclassifications reveals substantial differences between ED-GAT and transformer-based pretrained language models' performance.

Current research on preference classification is lacking and far from practical use. Real world comparisons can present characteristics that complicate the task, such as indirect comparisons, implicit comparisons, and ambiguous statements. The low F1-score of the existing state-of-the-art and noticeable differences in misclassifications across different models call for a more thorough research effort on preference classification in text.

2 Related work

Comparative sentence structures have been a subject of syntactic and semantic theories (Bresnan, 1973; Stechow, 1984; van Rooij, 2011). Early studies in computational linguistics include syntactic and semantic handling of comparative constructions (Rayner and Banks, 1988, 1990), comparative structures in question answering (Ballard, 1988), using quantifiers to identify comparisons (Friedman, 1989), and semantic interpretation of comparatives (Staab and Hahn, 1997).

Jindal and Liu (2006) present a binary classification dataset containing comparative and non-comparative sentences. They present a classifier based on Class Sequential Rules (CSR) and leverage comparative keywords to identify comparative sentences. Ganapathibhotla and Liu (2008) extend this work by annotating comparative sentences with the preferred entity.

Kessler and Kuhn (2014) annotate comparative sentences by identifying comparison predicates, entities being compared, aspect of comparison, and comparison type (gradable or non-gradable). However, they focus on reviews of only one product type (digital cameras) to create their dataset. Hence, their dataset lacks diversity in topics.

Panchenko et al. (2019) create CompSent-19, a cross-domain dataset for comparative argument mining. They propose a gradient boosting model based on pretrained sentence embeddings to identify the preferred entity. Ma et al. (2020) propose a model called Entity-aware Dependency-based Deep Graph Attention Network (ED-GAT) that consists of a multihop graph attention network with dependency relations to identify the preferred entity. The ED-GAT model achieves a micro F1-score of 87.43% on the CompSent-19 dataset.

Previous work on preference classification has overlooked implicit and indirect comparisons common in user-generated text such as app reviews. Further, existing datasets are either too small with a few comparative sentences or have a skewed distribution. For example, Ganapathibhotla and Liu's dataset contains only 837 comparative sentences, 84% of which have the first mentioned entity in the text as preferred. Only 15% of Kessler and Kuhn's dataset constitutes comparative sentences. Only 27% of the sentences in CompSent-19 (Panchenko et al., 2019) contain a preference, 70% of which prefer the first mentioned entity in the sentence.

Further, existing datasets consider the order of the appearance of compared entities in a sentence to annotate the preferred entity. For instance, annotations for CompSent-19 (Panchenko et al., 2019) and Ganapathibhotla and Liu's dataset are both determined based on the order of appearance of the entity in a sentence (i.e., is the first appearing entity in the sentence preferred or the second).

3 Method

We introduce the essential concepts below.

Comparative sentence : A sentence that contains information on similarity, dissimilarity, or preference between two entities.

Pixie includes (1) comparative sentences that lack comparative quantifiers, adjectives, or keywords, i.e., *indirect comparisons*, (2) *implicit comparisons* where only one of the compared entities is mentioned, and (3) *explicit comparisons* which mention both (including pronominal references).

Preferred entity : an entity that is chosen over another based on a stated or implied preference.

A preferred entity can be the CURRENT app (e.g., S_{1p} in Table 2), OTHER app (e.g., S_{2p} in Table 2), or NONE (i.e., ambiguous or no preference, e.g., S_{4p} in Table 2 or where non-gradable comparatives (such as *like*, *as . . .* , and *similar to*) link the entities, e.g., S_{3p} in Table 2).

	Sentence	Review
S_{1p}	***This app* is better than *Discover's app*.**	*Chase Mobile*
S_{2p}	I prefer the ***BBC app***.	*USA Today*
S_{3p}	Just as good as ***Uber*** app.	*Lyft*
S_{4p}	Makes me want to switch back to ***Pandora***, but ***it's*** just as bad.	*Spotify*

Table 2: Example sentences showing preference.

3.1 Dataset

We selected 179 popular apps on Apple App Store and collected their reviews. After some preliminary investigation, we excluded widely mentioned brand names such as Google, Microsoft, and Facebook, because they often appear in broader contexts than as a product. We removed app names synonymous with or formed of common words, such as Box (cloud storage) and Line (communication app) for higher precision in extracting comparative sentences. We were left with 141 apps, which we manually grouped into 23 genres, including *banking*, *airline*, and *communication*. Apps in the same genre are direct competitors. For example, *airline* apps include Delta, American, and United.

We extracted sentences that mention a competitor from each review and labeled each extracted sentence for comparison and preferred entity. When identifying mentions, we included common aliases or abbreviations on our name list, e.g., *Insta* for Instagram, *BA* for Bank of America, and *AA* for American Airlines to improve recall. Focusing on mentions of competitors ensures that Pixie includes indirect comparisons because such sentences are more likely to contain comparisons.

The dataset was annotated in three phases. In Phase 1, the authors annotated a sample dataset of 300 sentences based on an initial set of definitions and resolved any disagreements via discussions. We repeated this process for three iterations and produced annotation instructions for Phase 2. In Phase 2, each author annotated an equal number of sentences, and the disagreements were resolved by the first author, producing 4,793 annotated sentences. The interrater agreement (Krippendorff alpha) was 0.74 and 0.82 between the two annotators for comparison and preferred entity, respectively. Phase 3 involved crowdsourcing with 42 annotators, students in Natural Language Processing (NLP) course, each annotating around 400 sentences. 5,559 data points were labeled in Phase 3 with an interrater agreement (Krippendorff alpha) between the three annotators of 0.51 and 0.74 for comparison and preferred entity, respectively. We obtained the Institutional Review Board (IRB) approval for this task.

Once we removed duplicate and noncomparative sentences, we were left with 8,890 comparative sentences annotated for comparison type (IMPLICIT or EXPLICIT) and preferred entity (CURRENT, OTHER, or NONE). Table 3 shows the distribution of labels for each class in Pixie.

	Comparison Type		
Preferred Entity	Implicit	Explicit	Total
CURRENT	1,910	2,097	4,007
OTHER	2,199	1,069	3,268
NONE	758	857	1,615
Total	4,867	4,023	8,890

Table 3: Pixie Dataset Distribution.

To ensure that the dataset can be used to train a general-purpose preference classification model, we *mask* app mentions in each sentence. With no masking, the model may learn to differentiate between classes based on what users prefer more (app A or app B) in our dataset. Masking app mentions ensures that the model learns comparative and preference revealing linguistic structures and semantics instead of simply learning to differentiate between preferred entities in an exhaustive list of compared entities. We defined two tags for masking, *current_app* for the apps being reviewed and *other_app* for the competitor apps. App mentions are identified using the competitor app list for apps referred to by name, and pronoun references are substituted manually. Treating pronoun references as an explicit reference to app mentions ensures consistent based on our definitions, i.e., all explicit comparisons have two mentioned entities being compared, while all implicit comparisons have one. A portion of the dataset, $\approx$2,100 ($\sim$23.62%) sentences, had pronoun references that were resolved. Table 4 shows sentences masked for app mentions.

For a quick sanity check, whether Pixie contains indirect comparative sentences, we examine how many of the sentences in Pixie contain a comparative word. For this, we combine the list of opinion words from (Hu and Liu, 2004) and the list of comparative cue words from (Panchenko et al., 2019). Only 3,781 sentences (42.5% of Pixie) contain a comparative or opinion word showing that most of the sentences in Pixie lack comparative cues (i.e., are indirect comparisons).

Unlike prior datasets on preference classification (Ganapathibhotla and Liu, 2008; Panchenko et al., 2019), Pixie does not consider the order of appearance of compared entities for annotations. Pixie also offers a more balanced dataset than the existing ones for the task. For explicit comparisons (when both entities are present), 1909 sentences (47.45%) prefer entity that appears first, 1257 (31.25%) sentences prefer entity that appears later, and 857 sentences (21.30%) reveal no or mixed preference. Implicit comparisons mention only one entity so the order of appearance is irrelevant.

Pixie is publicly available[1] and contains original and masked sentences.

3.2 Experiments

Among traditional machine learning approaches, we experiment with AdaBoost (Hastie et al., 2009), Random Forest (Breiman, 2001) and Support Vector Machine (SVM) (Chang and Lin, 2011). We use SBERT (SentenceBERT) (Reimers and Gurevych, 2019) to obtain sentence embeddings for each masked sentence.

For transformer-based language models, we fine-tune variations of BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al., 2019) and XLNet (Yang et al., 2019). We experiment with BERT (Devlin et al., 2019), ALBERT (A Lite BERT) (Lan et al., 2019), and DeBERTa (Decoding-enhanced BERT with disentangled attention) (He et al., 2020). We fine-tune each model for 20 epochs using AdamW Optimizer with a learning rate of 5e-5 and a weight decay of 0.01. We use the train-test-validation split of 60-20-20.

We use segment embeddings to improve the performance of the transformer-based models. We assign different segment token ids to the competitor app (other_app) and the rest of the sentence to separate the entities being compared. We fine-tune pretrained models with segment embeddings along with token embeddings and attention masks.

To compare our results with ED-GAT, we convert the sentences in Pixie to follow the CompSent-19 format. Specifically, we add a token ([THIS]) for the current app in the front of each implicit sentence and map the labels CURRENT and OTHER to BETTER and WORSE, as applicable. NONE labels stay the same. We implemented ED-GAT with BERT embeddings and used eight GAT layers. We use the Hugging Face (Wolf et al., 2020) library for all transformer-based experiments.

To test the quality of Pixie, we run some cross-dataset experiments as well. We train a DeBERTa model on Pixie and test on CompSent-19 and vice-versa. Since the CompSent-19 dataset is highly skewed, we balanced both datasets to have the same train and test data split across all three classes via random oversampling with replacement. We keep all other model parameters and configurations the same and leverage the same number of samples for training and testing.

4 Results

Table 5 contains results for models trained and tested on Pixie. SVM achieves the highest weighted F1-score of 71.86% (among the traditional approaches), and DeBERTa (F1-score 83.34%), among transformer-based models.

Segment embeddings enhanced BERT and XLNet model's performance in terms of weighted average F1-scores, but a slight decline for DeBERTa's and ALBERT's performance.

The NONE and CURRENT classes consistently achieve the lowest and the highest F1-scores, respectively, for all models. The NONE class was also the most ambiguous class to annotate manually. Recall for the NONE class is lower than precision for all models except ED-GAT. All transformer-based models achieve a higher recall than precision for the CURRENT class except for ALBERT (without segment embeddings) and ED-GAT.

ED-GAT (Ma et al., 2020) trained on Pixie achieves a weighted average F1-score of 73.99%, with the highest F1-score (80.57%) for the CURRENT class and lowest (51.54%) for NONE.

Upon further analysis, we found that most of the incorrect classifications in transformer-based models are for the NONE class (71.64%), whereas, for ED-GAT, only 8.77% of the misclassified sentences belong to the NONE class. ED-GAT yielded most misclassifications for the CURRENT class (55.27% of misclassified instances) while only 14.93% of misclassifications for the transformer-based models belong to the CURRENT class.

Table 6 shows the results for the cross-dataset experiments. The weighted average F1-score improves by 4.08% with plain vanilla fine-tuning and 6.30% with segment embeddings when trained on Pixie and tested on CompSent-19. While the accuracy improves by 5.11% for plain vanilla fine-

[1]https://github.com/ahaque2/Pixie.git

	Original sentence	Masked sentence
1	*CNN* should leave journalism to the pros at *Fox* news.	<*current_app*> should leave journalism to the pros at <*other_app*> news.
2	way better than *Pandora* by a long shot!!!!	way better than <*other_app*> by a long shot!!!!
3	*This* is a great game just like *Temple run*	<*current_app*> is a great game just like <*other_app*>

Table 4: Original and masked comparative sentences.

Approach	Model	CURRENT			NONE			OTHER			WEIGHTED AVERAGE		
		Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Prior Work	ED-GAT	83.24	78.05	**80.57**	48.89	54.49	**51.54**	76.28	77.79	**77.03**	74.44	73.68	**73.99**
Traditional ML	AdaBoost	71.57	73.44	72.49	45.06	35.29	39.58	63.53	68.30	71.57	63.80	64.62	64.07
	Random Forest	71.27	80.42	75.57	71.31	26.93	39.10	64.98	74.73	69.52	68.97	68.62	66.72
	SVM	76.99	82.17	**79.49**	62.63	36.84	**46.39**	71.04	79.63	**75.09**	72.19	73.00	**71.86**
Transformer-Based	BERT	82.83	89.03	85.82	62.68	55.11	58.65	83.07	80.40	81.71	79.26	79.70	79.37
	DeBERTa	88.34	90.65	**89.48**	64.56	56.97	60.53	85.97	88.21	**87.07**	83.15	83.63	**83.34**
	ALBERT	87.83	87.28	87.55	61.37	60.99	**61.18**	84.70	85.60	85.15	81.87	81.89	81.88
	XLNet	83.45	90.52	86.84	67.06	52.32	58.78	83.99	84.38	84.19	80.67	81.33	80.77
Transformer-Based with Segment Embeddings	BERT	83.43	88.53	85.90	66.67	52.63	58.82	81.25	83.61	82.42	79.58	80.20	79.70
	DeBERT	88.31	91.40	**89.83**	64.34	56.97	**60.43**	85.35	86.52	**85.93**	82.87	83.35	**83.06**
	ALBERT	86.26	87.66	86.95	65.92	54.49	59.66	81.90	87.29	84.51	80.96	81.50	81.10
	XLNet	85.68	90.27	87.92	61.86	55.73	58.63	85.51	84.07	84.79	81.29	81.72	81.45

Table 5: Results (in %) for preference classification on Pixie. Bold indicates highest F1-scores for each category.

Approach	Fine-tuning	Testing	Prec	Recall	F1	Accuracy
Plain vanilla	CompSent-19	Pixie	65.46	59.89	59.23	59.89
Plain vanilla	Pixie	CompSent-19	65.19	65.00	63.31	65.00
With segment embeddings	CompSent-19	Pixie	67.84	59.44	57.70	59.44
With segment embeddings	Pixie	CompSent-19	66.07	65.72	64.00	65.72

Table 6: Results for cross-dataset experiments. The values are in %.

tuning and 6.28% with segment embeddings. The improvement primarily is in the recall, demonstrating that Pixie includes more diverse comparative sentences than CompSent-19.

5 Conclusion

Masking compared entities ensure that Pixie can be used to train a general-purpose preference classification model. Additional analysis is needed to claim the domain generality of our dataset—that is, whether a model trained on Pixie can identify the preferred entity in texts from other domains such as scientific papers and news. Comparative sentences in Pixie are limited to user-generated text and may not generalize well over more formal texts.

Both BERT and XLNet show improvements with segment embeddings, suggesting that the demarcation of the other app helps the model identify the preferred entity. The traditional machine learning models perform worst and the transformer-based pretrained models fine-tuned on Pixie achieve a substantially better performance than the state-of-the-art approaches for preference classification.

Identifying preferences in user reviews can aid developers in understanding user expectations about mobile apps. Users often express their likes and dislikes about an app or feature by comparing it with alternative apps and features. Understanding user preferences can be particularly valuable in enhancing the functionality as well as security and privacy features of apps. A user's preferences regarding apps would depend not only on how well the app is constructed relative to its competitors but also on how easily the app is used by end-users. For example, security concerns may be signaled by descriptions of steps to access sensitive financial or medical data (Guo and Singh, 2020) expressed in association with comparisons. A follow-on direction is to extract and prioritize user expectations by identifying the specific features of an app of greatest influence on the indirect or direct comparisons in a review.

Acknowledgments

This research is supported by the US Department of Defense under the Science of Security Lablet (SoSL) grant to NC State University. We thank the ACL reviewers for their helpful comments.

References

Bruce W. Ballard. 1988. A general computational treatment of comparatives for natural language question answering. In *Proceedings of the 26th Annual Meeting on Association for Computational Linguistics*, ACL, page 41–48, USA. Association for Computational Linguistics.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Joan W. Bresnan. 1973. Syntax of the comparative clause construction in English. *Linguistic Inquiry*, 4(3):275–343.

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3).

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 17th Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ronen Feldman, Moshe Fresko, Jacob Goldenberg, Oded Netzer, and Lyle Ungar. 2007. Extracting product comparisons from discussion boards. *Proceedings - IEEE International Conference on Data Mining, ICDM*, 8001:469–474.

Carol Friedman. 1989. A general computational treatment of the comparative. In *27th Annual Meeting of the Association for Computational Linguistics*, pages 161–168, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Murthy Ganapathibhotla and Bing Liu. 2008. Mining opinions in comparative sentences. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING, page 241–248, USA. Association for Computational Linguistics.

Hui Guo and Munindar P. Singh. 2020. Caspar: Extracting and synthesizing user stories of problems from app reviews. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, ICSE, page 628–640, New York, NY, USA. Association for Computing Machinery.

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. 2009. Multi-class AdaBoost. *Statistics and its Interface*, 2(3):349–360.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. DeBERTa: Decoding-enhanced BERT with disentangled attention. *arXiv preprint*, abs/2006.03654.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD, page 168–177, New York, NY, USA. Association for Computing Machinery.

Nitin Jindal and Bing Liu. 2006. Identifying comparative sentences in text documents. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR, page 244–251, New York, NY, USA. Association for Computing Machinery.

Wiltrud Kessler and Jonas Kuhn. 2014. A corpus of comparisons in product reviews. In *Proceedings of the 9th International Conference on Language Resources and Evaluation (LREC)*, pages 2242–2248, Reykjavik, Iceland. European Language Resources Association (ELRA).

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint*, abs/1909.11942.

Yuanchun Li, Baoxiong Jia, Yao Guo, and Xiangqun Chen. 2017. Mining user reviews for mobile app comparisons. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 1(3).

Nianzu Ma, Sahisnu Mazumder, Hao Wang, and Bing Liu. 2020. Entity-aware dependency-based deep graph attention network for comparative preference classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5782–5788, Online. Association for Computational Linguistics.

Walid Maalej and Hadeer Nabil. 2015. Bug report, feature request, or simply praise? On automatically classifying app reviews. In *Proceedings of the 23rd IEEE International Requirements Engineering Conference (RE)*, pages 116–125, Ottawa, ON, Canada. IEEE Press.

Fabio Palomba, Mario Linares-Vásquez, Gabriele Bavota, Rocco Oliveto, Massimiliano Di Penta, Denys Poshyvanyk, and Andrea De Lucia. 2015. User reviews matter! Tracking crowdsourced reviews to support evolution of successful apps. In *Proceedings of the 31st IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 291–300, Bremen, Germany. IEEE Press.

Alexander Panchenko, Alexander Bondarenko, Mirco Franzek, Matthias Hagen, and Chris Biemann. 2019. Categorizing comparative sentences. In *Proceedings* *of the 6th Workshop on Argument Mining*, pages 136–145, Florence, Italy. Association for Computational Linguistics.

Manny Rayner and Amelie Banks. 1988. Parsing and interpreting comparatives. In *26th Annual Meeting of the Association for Computational Linguistics*, pages 49–60, Buffalo, New York, USA. Association for Computational Linguistics.

Manny Rayner and Amelie Banks. 1990. An implementable semantics for comparative constructions. *Computational Linguistics*, 16(2):86–112.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Steffen Staab and Udo Hahn. 1997. Comparatives in context. In *Proceedings of the 14th National Conference on Artificial Intelligence*, AAAI/IAAI, page 616–621, Providence, Rhode Island. AAAI Press.

Arnim Von Stechow. 1984. Comparing semantic theories of comparison. *Journal of Semantics*, 3(1-2):1–77.

Robert van Rooij. 2011. Implicit versus explicit comparatives. In *Vagueness and Language Use*, pages 51–72. Palgrave Macmillan UK, London.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R. Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32, Vancouver. Curran Associates, Inc.