StanceMining: An open-source stance detection library supporting time-series and visualization

Benjamin D. Steel
McGill University
benjamin.steel@mail.mcgill.ca

Derek Ruths
McGill University
derek.ruths@mcgill.ca

Abstract

Despite the size of the field, stance detection has remained inaccessible to most researchers due to implementation barriers. Here we present a library that allows easy access to an end-to-end stance modelling solution. This library comes complete with everything needed to go from a corpus of documents, to exploring stance trends in a corpus through an interactive dashboard. To support this, we provide stance target extraction, stance detection, stance time-series trend inference, and an exploratory dashboard, all available in an easy-to-use library. We hope that this library can increase the accessibility of stance detection for the wider community of those who could benefit from this method.

1 Introduction

The field of stance detection —the identification of the attitude of a document author to a target, as represented by a topic, claim, entity, etc. (Mohammad et al., 2016) —has produced a number of methods critical to the understanding of social behaviour. However, it remains a method that requires a committed natural language processing (NLP) expert to apply. While other NLP fields have successfully made their technology available for general practitioners, with topic modelling being a prime example, stance detection remains off limits to general use. Beyond this, stance detection has thus far focused on the situation of having a set of documents with pre-defined stance targets (the idea or issue a stance is expressed on, here in the form of noun-phrases or claims, as used previously (Zhao and Caragea, 2024)). We present a library that uses a combination of new and prior methods to allow a user to go from a raw corpus of documents, to an organised set of stance targets and stance target trends, with little-to-no tuning needed.

Stance detection is frequently used in a temporal context to understand how attitudes are changing over time. In prior work, the outputs have been naively assembled into a time series using a moving average (Introne, 2023; Almadan et al., 2023). We account for the error in the stance classifier and the noisiness of the data by using Gaussian processes (GPs) with a custom likelihood to model the temporal trends of stance. We show an explanation of this problem in Fig. 1.

Our library contains an easy-to-deploy web-app that allows for the exploration of the features output from our library. In addition, it comes with small fine-tuned models and defaults that allow it to run on consumer-grade GPUs [1], making it accessible to researchers with modest compute budgets. We have seen the value that accessible topic modelling has provided to the larger community that can benefit from using topic modelling but does not have the technical capacity to implement their own topic models, and we hope that, similarly, this library can benefit the larger community of social scientists who have much to gain from easily accessible stance detection.

We present two novel contributions: First, to our knowledge, no stance detection method is available in a library/package form, only as research repositories specific to a context and dataset. We go beyond this and release a library that is designed to be generally usable. Second, no prior work has produced a method that can take stance labels with timestamps, and infer a continuous time-varying stance, considering the error of the classifier.

We release this library under an MIT license at https://github.com/bendavidsteel/stancemining. Scripts to reproduce our results are available in https://github.com/bendavidsteel/stancemining/tree/main/experiments. In addition, we present a video demonstrating the system at this link: https://youtu.be/4tvqq8GTUHU.

[1] Here defined as having less than 16GB VRAM

Proceedings of The 14th International Joint Conference on Natural Language Processing and The 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics: System Demonstrations, pages 67–76

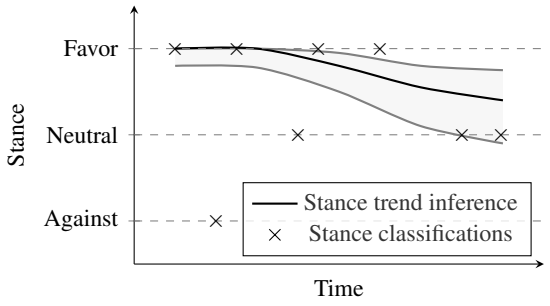


Figure 1: Given the ordinal stance classifications as ×, with labels 'Favor', 'Neutral', and 'Against', arrayed in time, how should we infer the latent stance? We propose to use a Gaussian process with a customized ordinal likelihood to infer the latent stance trend. This will allow us to infer the latent stance from the stance classifications as shown in the figure - that is, in a smoothly varying manner that balances fitting signal and avoiding noise. This allows us to factor in the error of the stance detection classifier into our inference, alongside setting a prior on the extent to which the stance trend will vary, allowing us to ignore noise.

2 Background

Services exist to provide stance-detection-like models to a practitioner audience, but they are either proprietary (sum) or domain specific (Stab et al., 2018). Methods for inferring stance trends from stance observations have been limited to rolling averages (Introne, 2023), or aggregation on a time interval basis (Almadan et al., 2023). We produce models that can both interpolate, and consider the error of the classifier.

For stance detection output visualization, previous work has produced solutions (Wu et al., 2014; Martins et al., 2017; Kucher et al., 2020, 2016), but all are either not open-source/publicly available, or are specific to a particular domain, or both.

3 Implementation

We depict the system in Fig. 2, showing the functionality that the library affords. By default, for cases where the specific stance-targets of a corpus are not known a priori, the fine-tuned stance target extraction model will extract stance targets from each document. The fine-tuned stance detection models will then find the stance of each document on each stance target mapped to that document. Additional stance targets can optionally be discovered via clustering, using the method detailed in **?**. Alternately, pre-defined stance targets can be provided, in which case the library will find the stance of each document on each pre-defined stance targets.

For out-of-the-box use, we use our two fine-tuned models by default, hosted on HuggingFace model storage to allow distribution. In practice, stance detection frequently needs custom models for domain specific data, so the library allows using any local transformers compatible fine-tuned model. We train our base stance extraction model on VAST (Allaway and Mckeown, 2020) and EZ-STANCE noun-phrase (Zhao and Caragea, 2024). Stance targets represented by claims are popular in stance detection (Küçük and Can, 2020; Zhao and Caragea, 2024), so we additionally provide fine-tuned claim extraction models. This choice of noun-phrases or claims for extracted stance targets is configurable by the user in the library.

Since cross-dataset generalization in stance detection is poor (Ng and Carley, 2022; Zhao and Caragea, 2024), we fine-tuned a modern small language model on several datasets to produce a model that has more generalizability. While this comes at the risk of each dataset's slightly different definition of stance distorting the learned signal, it should improve the generalizability of the model. Specifically, we use the following datasets: SemEval Task 6 dataset (Mohammad et al., 2016), VAST (Allaway and Mckeown, 2020), EZ-STANCE noun-phrase and claim datasets (Zhao and Caragea, 2024), P-STANCE (Li et al., 2021), and the multi-turn conversational stance detection datasets MT-CSD (Niu et al., 2024) and CTSDT (Li et al., 2023). The use of the multi-turn datasets means that our library supports documents with contextual threads, common in media data. Corresponding to the datasets we select, the specific form of stance detection we focus on is topic/entity stance detection (Zhu et al., 2025).

We use 16-bit models to maintain high throughput on older GPUs [2]. We use vLLM (Kwon et al., 2023) for fast LLM inference. This enables processing of a dataset of ∼1300 posts in 4 minutes.

Stance detection using inputs from audio data, whether from social media videos or podcasts, is a common use-case. We therefore provide helper functions to transcribe audio and video files, using WhisperX (Bain et al., 2023) and pyannote (Plaquet and Bredin, 2023). Our library does not currently support image/video inputs.

Stance Trend Inference GP models use a base model that outputs a set of Gaussian distributions corresponding to input points, and a likelihood, that

[2] https://docs.vllm.ai/en/latest/features/quantization/supported_hardware.html

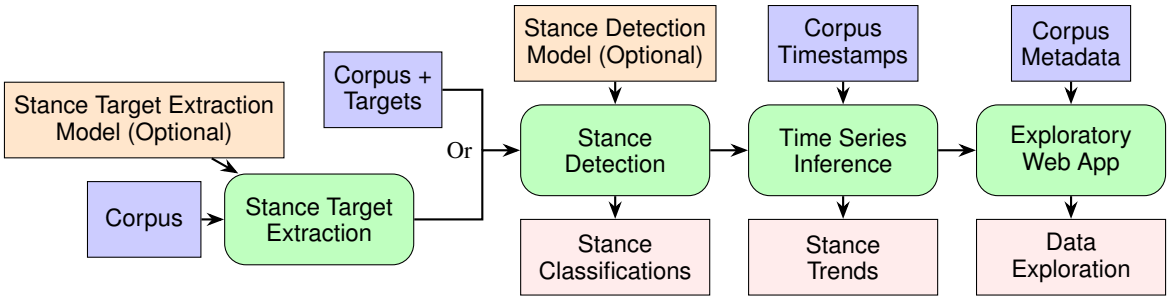


Figure 2: System diagram of *stancemining*. Origin items in blue are inputs to the system, origin items in orange are optional models (*stancemining* provides these models by default), intermediate items in green are components of the *stancemining* library, and endpoint items in red are outputs from the library.

takes as input samples from the base model, and defines how those signals correspond to the actual data seen. In our case, the base model is modelling the function between time and true latent stance, and the likelihood is modelling the relationship between the true latent stance and the observed stance classifications. Stance classifications are ordinal in nature: a 'neutral' post favors a target more than an 'against' post, and an 'favor' post favors a target more than a 'neutral' post. We therefore use an ordinal likelihood (Chu et al., 2005), and model the stance as a continuous time-varying value between -1 ('against') and 1 ('favor').

There are two conditions specific to our use-case. First and most importantly, stance detection is done with classification models that make errors. If a stance detection model is better at classifying 'favor' posts than 'against' posts, we will have a systematic error in our trend inferences if we do not account for this error. We can estimate the error of the classifier from its performance on an evaluation set. The ordinal likelihood function for the 3 true stance labels would generally be defined as:

$$p(Y = \text{Fav.}|F) = \phi\left(\frac{a_0 - F}{\sigma}\right)$$
$$p(Y = \text{Neu.}|F) = \phi\left(\frac{a_1 - F}{\sigma}\right) - \phi\left(\frac{a_0 - F}{\sigma}\right)$$
$$p(Y = \text{Aga.}|F) = 1 - \phi\left(\frac{a_1 - F}{\sigma}\right)$$

where ϕ is the cumulative density function of a Gaussian (the inverse probit function), σ is a parameter to be learned, the data are integer values from 0 to k, and the bin edges where the labels switch are $[a_0, a_1]$, which we set to $[-0.5, 0.5]$

However, we want to model the observed probabilities. So first we normalize the confusion matrix $\mathbf{C}$ such that each row sums to 1: $\sum_{j=1}^{3} c_{ij} = 1$ for all $i \in 1, 2, 3$. We then multiply the true stance probability vector $p_S = [p(Y = \text{Fav.}|F), p(Y = \text{Neu.}|F), p(Y = \text{Aga.}|F)]$ with the normalized confusion matrix to obtain the final observed stance probabilities $\hat{p}_S = Cp_S$, which are used as the probabilities of the likelihood categorical output.

The second condition specific to our use-case is that the stance should be clamped between -1 and 1. Without this, for an array of 'favor' observations, the GP model could reasonably infer a stance of any value over 1, whereas we want to limit the stance at 1 for the sake of trend comparison. We model this by inferring the inverse hyperbolic tangent with the GP base model, and then transform the output of this model for predictions and the likelihood by applying a hyperbolic tangent transform.

We use a GP model with constant mean and the radial basis function for the covariance kernel, with a log-normal prior for the lengthscale parameter. In this case, we are inferring the time-varying trend of a person's stance on an issue (as opposed to an organization etc.), so we choose a lengthscale prior of $\mu = 2, \sigma = 0.1$ based on prior work studying the correlation of user attitudes over time in Krosnick (1988). We use the ordinal likelihood developed by Chu et al. (2005), and optimize the model using stochastic variational inference, using natural gradient descent (Salimbeni et al., 2018).

A critical aspect of getting the model to train fast and effectively is the learning rates of the natural gradient optimizer, and the learning rate and learning rate scheduler of the hyperparameter and likelihood parameter optimizer. We run hyperparameter sweeps of these three variables in Section C. To implement the model, we use GPyTorch (Gardner et al., 2018). To improve training speed, we add a number of optimizations, including compiling

the model using PyTorch [3], converting functions to TorchScript, and a batching process which trains models of comparable size in parallel.

When calling the infer trends function on the library, the user can specify filter columns. These are columns in the dataframe where trends should be calculated for each unique value. This is useful for when a user wants to calculate time series trends for different users, sources, accounts etc.

4 Application Use

Once stance features have been extracted from the corpus using *stancemining*, it is useful to be able to quickly explore them using an interactive application. We therefore include a ready-to-use web-app for this purpose. This web-app can be easily deployed via Docker Compose for any user data via the use of environment variables, as long as the data has been saved in the format and structure specified in the documentation. There is an option to enable authentication in the application if necessary for sensitive data. The backend of the application uses FastAPI [4], and the frontend uses React [5].

The user is presented with two main views in the app: a stance target map view, and a timeline view. The map view (Fig. 3a) shows a 2-dimensional map of stance targets, where stance targets are embedded using the 'GIST-small-Embedding-v0' text embedding model (Solatorio, 2024), those embeddings are reduced to 2-dimensions using UMAP (McInnes et al., 2018), and plotted as a scatter plot. Points are coloured with the mean stance of documents on that target, and hovering over the point shows the proportions of each stance on that target. This plot allows the user to explore stance targets in a 2D semantic space, and the discovery of more stance targets than semantic search alone. If a user clicks on a point, they are shown the temporal trends of that target in the timeline view.

The timeline view (Fig. 3b) shows the inferred trend mean for each stance target, alongside its confidence intervals as computed by the GP model. When we load the trend data, the backend automatically finds filter types, and allows the user to display trends broken down by filter value side by side, or one at a time. This is useful when breaking down stance target trends into individual users or sources.

We summarise the full use of the *stancemining* library and web app in the following steps. While each component of the system can be used separately, this sequence represents its full potential:

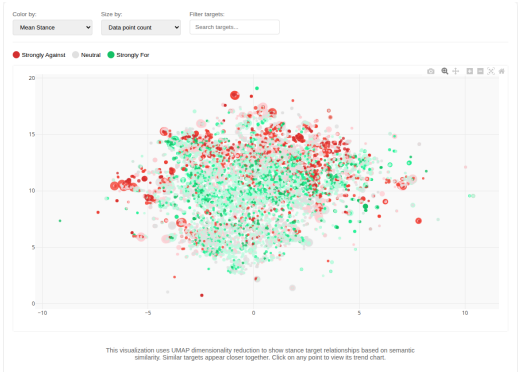
1. **Extract Targets (Optional)**: Extract stance targets (noun-phrases or claims) from corpus, using default model or custom model if there are specific domain requirements.
2. **Stance Detection**: Detect stance of documents on targets using default models, or custom models if there are specific domain requirements.
3. **Stance Trend Inference**: Do stance trend inference if corpus has timestamps.
4. **Load Web App**: Load web app using saved *stancemining* outputs, specify options via environment variables.
5. **Explore Target Map**: Explore stance targets in the target map view.
6. **View Target Trend**: Zoom in on a specific target timeline, filter by metadata attributes.

5 Evaluation

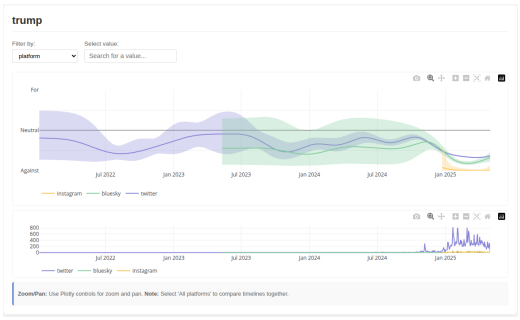
Where possible, we evaluate components of our system against prior work. Otherwise, we provide multiple competing methods to add context to the range of possible metrics. We did not do any formal evaluation of the web application with end users.

Stance Target Extraction We evaluate the model using BERTScore (Zhang et al., 2019) and BLEU (using the SacreBLEU package) (Post, 2018), and show our results in Table 1. We compare to the smallest open-source model evaluated in Akash et al. (2024). No other direct task and dataset comparisons exist. For the task of noun-phrase stance target extraction, we fine-tune our models on a combined dataset of the document - stance target noun phrases from VAST (Allaway and Mckeown, 2020) and the noun-phrase targets of EZ-STANCE (Zhao and Caragea, 2024). We do not compare against prior work here, as previous work has only evaluated one extracted stance target per document (Akash et al., 2024), as opposed to extracting multiple targets per document. For these tasks we use Qwen 3 models (Yang et al., 2025) and SmolLM2 models (Allal et al., 2025), with specific sizes reported in the tables. For our claim extraction models, we fine-tune them on the document - stance target claims pairs from EZ-STANCE (Zhao and Caragea, 2024). We show metrics from this fine-tuning in Table 2. No comparable results are available for this task on this dataset to our knowledge.

[3] https://pytorch.org/
[4] https://fastapi.tiangolo.com/
[5] https://react.dev/



(a) 2-dimensional map view of stance targets.



(b) Trend timelines view of a stance target broken down by filter values.

Figure 3: Two screenshots from the *stancemining* dashboard.

Model	Num. params.	EZ-STANCE BERTScore F1	EZ-STANCE BLEU F1	VAST BERTScore F1	VAST BLEU F1
Llama-3-8B (Akash et al., 2024)	8B	0.78	-	0.84	-
Qwen3-0.6B (ours)	752M	0.90	0.67	0.91	0.33
SmolLM2-360M-Instruct (ours)	360M	0.90	0.67	0.92	0.20

Table 1: Noun-phrase stance target extraction evaluation results. We compare to the smallest open-source model evaluated in Akash et al. (2024). No other direct task and dataset comparisons exist.

Model	Num. params.	EZ-STANCE BERTScore F1	EZ-STANCE BLEU F1
Qwen3-1.7B (ours)	2.03B	0.89	0.16
Qwen3-0.6B (ours)	752M	0.88	0.04

Table 2: Claim stance target extraction evaluation. No comparable results are available for this task on this dataset to our knowledge.

Stance Detection We evaluate our fine-tuned stance detection models using the macro F1 score, as is typical in stance detection (Zhao and Caragea, 2024; Allaway and Mckeown, 2020), and show our results in Table 3. We experimented with several hyperparameters and report the best results here, using 4 epochs, $batch\_size \times grad\_accumulation\_size = 32$, a learning rate of 0.0001, and classifying using a classification head (instead of using causal language modelling to generate the label). We report other hyperparameters experimented with in App. A. In the table we also highlight the number of parameters in each model, given that part of the aim of *stancemining* is to make it accessible, thereby necessitating small models that work with limited resources.

Stance Trend Inference To test the GP model, we use synthetic data. This also allows us to model the relationship between a true latent stance and stance classifications across time, lacking a dataset for fine-grained stance over time. We detail our synthetic data generation process in Section B. We use 3 parameters to parameterize our synthetic data, the number of observations n_d, the random walk scale σ_{walk} which determines how much the latent stance trend varies, and σ_{noise} which determines the amount of noise in the stance expressions (i.e. someone who consistently favors a stance target or producing varied views on it).

To improve training time and model performance, we ran hyperparameter sweeps on the learning rates of the natural gradient optimizer for the GP model, and the learning rate and scheduler for the likelihood parameters and model hyperparameters. We averaged across several synthetic data configurations, including n_d, σ_{noise}, and σ_{walk}. We determine scheduler performance by averaging across all learning rate values, and then sorting by minimum loss achieved in 1000 epochs. We detail this experiment in Sec. C. Using no scheduler achieves the best loss, but a cosine scheduler achieves the fastest convergence. We therefore use no learning rate scheduler in our library. When using no learning rate scheduler, the most effective learning rate was 0.2 for both learning rates.

Next, we evaluate our method of inferring stance trends against other regression methods. Other methods have used a rolling average to smooth user stance observations (Introne, 2023) or simply

Model	Num. params.	$F1_{macro}$ SemEval	VAST	EZ-STANCE	EZ-STANCE claim	P-STANCE	MT-CSD	CTSDT
TGA Net	111M	-	0.665	-	-	-	-	-
BART-MNLI-e_p	205M	-	-	0.669	0.885	-	-	-
COLA	Prop.	-	0.73	-	-	-	-	
SmolLM2-135M-Instruct (Ours)	135M	0.57	0.71	0.61	0.81	0.78	0.56	0.67
SmolLM2-360M-Instruct (Ours)	360M	0.59	0.75	0.64	0.85	0.82	0.60	0.64

Table 3: Stance detection F1 scores for each dataset. With models BART-MNLI-e_p (Zhao and Caragea, 2024), TGA Net (Allaway and Mckeown, 2020), and COLA (Lan et al., 2024) (COLA uses GPT 3.5 Turbo, hence the proprietary label in the number of parameters cell.) (While COLA evaluates on P-Stance and SemEval, they report separate macro F1s for each target, making comparison here difficult).

aggregate user stance observations on a time period basis (Almadan et al., 2023), but we consider it very useful to be able to interpolate the values, and as such, we only use methods capable of interpolation here. We evaluate LOWESS (Cleveland, 1979) and a spline model (De Boor and De Boor, 1978). Specifically, we found using a cubic smoothing spline with ridge regression with $\alpha = 0.1$ as a regularization parameter to produce the best data fits. We evaluated each method at 10 values of σ_{noise} between 0.05 and 0.5, 10 values of σ_{walk} between 0.005 and 0.05, and n_d randomly sampled from the range $(5, 30)$ (Detailed in App. D). Our GP method consistently obtains lower mean squared error (MSE) values than the other two methods evaluated. To obtain the overall MSEs, we simply take all MSEs measured for each point in the noise and random walk scales, and obtain their mean. We show these results, plotted against training time, in Fig. 4, showing that our GP method obtains a better mean MSE overall compared to the two methods.

However, this is not without a cost in training time: the process of training and obtaining predictions from GPs is much slower than for LOWESS and splines. This training time is acceptable when we want to have confidence in the stance trends we obtain for further modelling, but we provide LOWESS as a faster alternative in the library for those who want faster stance trend inference. Actual numbers are in Appendix D.

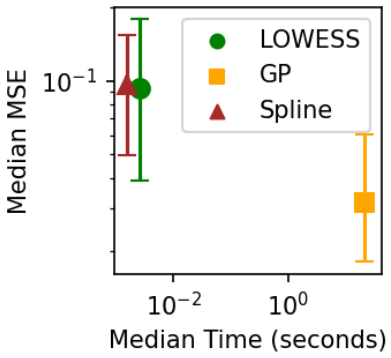


Figure 4: Comparison of median training time and median MSE between our GP, LOWESS, and splines over 100 runs with varying synthetic data parameters. Error bars indicate quartiles. Error bars for runtime are not visible as the runtime variance is small relative to marker size. Our GP method outperforms the other methods, but at the cost of increased training time.

6 Conclusion

We developed a library that allows a user to provide a document corpus expressing stance on unknown issues, optionally with associated timestamps and metadata, and detect the stance of the texts on the discovered targets. We believe that —just as easy-to-use topic modelling tools have enabled widespread use of these tools where they were previously inaccessible —easy-to-access stance detection can unlock new experimental approaches for groups where stance detection was previously inaccessible. We hope this tool unlocks improved understanding of opinion and attitude processes for a larger community of social scientists.

7 Ethical Considerations

Stance detection enables inference of attitudes from unconsenting text authors on issues they discuss, which is privacy-violating. It also allows greater insight into social processes, allowing researchers to work towards understanding social processes. However, the current nature of stance detection is that it requires dedicated work from an NLP engineer to implement, in addition to compute resources to train models, meaning that only large incumbents can use it. Our library aims to democratize access to stance detection, such that groups with fewer resources can use it in their studies, while larger groups were always able to use it.

References

SUMMITX. https://www.summetix.com/. Accessed: 2025-09-10.

Abu Ubaida Akash, Ahmed Fahmy, and Amine Trabelsi. 2024. Can large language models address open-target stance detection? *arXiv preprint arXiv:2409.00222*.

Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo, Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav, and 1 others. 2025. Smollm2: When smol goes big–data-centric training of a small language model. *arXiv preprint arXiv:2502.02737*.

Emily Allaway and Kathleen Mckeown. 2020. Zero-shot stance detection: A dataset and model using generalized topic representations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8913–8931.

Ali Almadan, Mary Lou Maher, and Jason Windett. 2023. Stance detection for gauging public opinion: A statistical analysis of the difference between tweet-based and user-based stance in twitter. In *Future of Information and Communication Conference*, pages 358–374. Springer.

Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. 2023. Whisperx: Time-accurate speech transcription of long-form audio. *INTERSPEECH 2023*.

Wei Chu, Zoubin Ghahramani, and Christopher KI Williams. 2005. Gaussian processes for ordinal regression. *Journal of machine learning research*, 6(7).

William S Cleveland. 1979. Robust locally weighted regression and smoothing scatterplots. *Journal of the American statistical association*, 74(368):829–836.

Carl De Boor and Carl De Boor. 1978. *A practical guide to splines*, volume 27. springer New York.

Jacob Gardner, Geoff Pleiss, Kilian Q Weinberger, David Bindel, and Andrew G Wilson. 2018. Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. *Advances in neural information processing systems*, 31.

Joshua Introne. 2023. Measuring belief dynamics on twitter. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 17, pages 387–398.

Jon A Krosnick. 1988. Attitude importance and attitude change. *Journal of Experimental Social Psychology*, 24(3):240–255.

Kostiantyn Kucher, Rafael M Martins, Carita Paradis, and Andreas Kerren. 2020. Stancevis prime: visual analysis of sentiment and stance in social media texts. *Journal of Visualization*, 23(6):1015–1034.

Kostiantyn Kucher, Teri Schamp-Bjerede, Andreas Kerren, Carita Paradis, and Magnus Sahlgren. 2016. Visual analysis of online social media to open up the investigation of stance phenomena. *Information Visualization*, 15(2):93–116.

Dilek Küçük and Fazli Can. 2020. Stance detection: A survey. *ACM Computing Surveys (CSUR)*, 53(1):1–37.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Xiaochong Lan, Chen Gao, Depeng Jin, and Yong Li. 2024. Stance detection with collaborative role-infused llm-based agents. In *Proceedings of the international AAAI conference on web and social media*, volume 18, pages 891–903.

Yingjie Li, Tiberiu Sosea, Aditya Sawant, Ajith Jayaraman Nair, Diana Inkpen, and Cornelia Caragea. 2021. P-stance: A large dataset for stance detection in political domain. In *Findings of the association for computational linguistics: ACL-IJCNLP 2021*, pages 2355–2365.

Yupeng Li, Dacheng Wen, Haorui He, Jianxiong Guo, Xuan Ning, and Francis CM Lau. 2023. Contextual target-specific stance detection on twitter: Dataset and method. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 359–367. IEEE.

Rafael Martins, Vasiliki Simaki, Kostiantyn Kucher, Carita Paradis, Andreas Kerren, and 1 others. 2017. Stancexplore: Visualization for the interactive exploration of stance in social media. In *2nd Workshop on Visualization for the Digital Humanities*.

Leland McInnes, John Healy, and James Melville. 2018. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*.

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. Semeval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th international workshop on semantic evaluation (SemEval-2016)*, pages 31–41.

Lynnette Hui Xian Ng and Kathleen M Carley. 2022. Is my stance the same as your stance? a cross validation study of stance detection datasets. *Information Processing & Management*, 59(6):103070.

Fuqiang Niu, Min Yang, Ang Li, Baoquan Zhang, Xiaojiang Peng, and Bowen Zhang. 2024. A challenge dataset and effective models for conversational stance detection. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 122–132.

Alexis Plaquet and Hervé Bredin. 2023. Powerset multi-class cross entropy loss for neural speaker diarization. In *Proc. INTERSPEECH 2023*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. 2018. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In *International Conference on Artificial Intelligence and Statistics*, pages 689–697. PMLR.

Aivin V. Solatorio. 2024. Gistembed: Guided in-sample selection of training negatives for text embedding fine-tuning. *arXiv preprint arXiv:2402.16829*.

Christian Stab, Johannes Daxenberger, Chris Stahlhut, Tristan Miller, Benjamin Schiller, Christopher Tauchmann, Steffen Eger, and Iryna Gurevych. 2018. ArgumenText: Searching for arguments in heterogeneous sources. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 21–25, New Orleans, Louisiana. Association for Computational Linguistics.

Yingcai Wu, Shixia Liu, Kai Yan, Mengchen Liu, and Fangzhao Wu. 2014. Opinionflow: Visual analysis of opinion diffusion on social media. *IEEE transactions on visualization and computer graphics*, 20(12):1763–1772.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

Chenye Zhao and Cornelia Caragea. 2024. Ez-stance: A large dataset for english zero-shot stance detection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15697–15714.

Zhengyuan Zhu, Zeyu Zhang, Haiqi Zhang, and Chengkai Li. 2025. Ratsd: Retrieval augmented truthfulness stance detection from social media posts toward factual claims. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 3366–3381.

A Stance Detection Training

We tested several models and hyperparameters to improve our final fine-tuned stance detection model, results shown in Table 4. In some cases where epochs are low we aborted training early due to low evaluation set metrics.

B Synthetic Data Generation

Then sample the observed post stances by sampling them from categoricals indexed by the true quantized post stances, using categorical probabilities obtained by normalizing the confusion matrix of the stance detection classifier on the test set.

We start with our timestamps representing each day in a year of 365 days.

$$t = [0, 1, 2, ..., 365]$$

We then sample the stance on the first day.

$$z_0 \sim \mathcal{U}(-1, 1)$$

Then simulate a random walk for each timestamp.

$$z_i = max(min((\mathcal{N}(z_{i-1}, \sigma^2_{\text{walk}}), -1), 1)$$

$$i = 1, 2, \ldots, n_{\text{time}} - 1$$

We then draw n_{obs} elements from the vectors t and z to serve as our observations t_{obs} and z_obs. We model diversity of stance expression (i.e. someone in favor of something will sometimes say things neutral or even against that thing) by sampling the latent user post stance values from normal distributions centred at the true stance, with scale σ_{noise}:

$$y_i \sim \mathcal{N}(z_i, \sigma_{noise}) \forall i$$

We then clamp these values between -1 and 1, and round them to the nearest integer to simulate the quantizing nature of classification.

$$s_i = round(min(max((y_i, -1), 1)) \forall i$$

where $s_j \in \{-1, 0, 1\}$

Then, using the normalized confusion matrix of the classifier, let $P_{k,\ell}$ be the probability of predicting class ℓ given true class k:

$$P = \begin{pmatrix} P_{-1,-1} & P_{-1,0} & P_{-1,1} \\ P_{0,-1} & P_{0,0} & P_{0,1} \\ P_{1,-1} & P_{1,0} & P_{1,1} \end{pmatrix}$$

where each row sums to 1: $\sum_{\ell \in \{-1,0,1\}} P_{k,\ell} = 1$

We get the classification probabilities as indexed by the true latent classifications to obtain the observed classifications:

$$\hat{s}_j \sim \text{Categorical}(P_{s_j,\cdot}), \quad j = 1, 2, \ldots, n_{\text{obs}}$$

Model Name	Epochs	Grad Accum Steps	Batch Size	LR.	Classification Method	$F1_{macro}$
SmolLM2-360M-Instruct	4	8	4	1.0e-4	head	0.744
SmolLM2-360M-Instruct	2	8	4	1.0e-4	head	0.738
SmolLM2-360M-Instruct	1	8	4	1.0e-4	head	0.724
SmolLM2-135M-Instruct	4	4	8	1.0e-4	head	0.715
SmolLM2-135M-Instruct	8	4	8	1.0e-4	head	0.709
SmolLM-135M-Instruct	4	4	8	1.0e-4	head	0.708
SmolLM2-135M-Instruct	4	4	8	1.0e-4	head	0.708
SmolLM2-135M-Instruct	4	4	8	1.0e-4	head	0.707
SmolLM-135M-Instruct	8	8	8	1.0e-4	head	0.705
SmolLM-135M-Instruct	8	4	8	5.0e-5	head	0.703
SmolLM-135M-Instruct	1	4	8	1.0e-4	head	0.651
SmolLM2-135M	2	8	4	1.0e-4	head	0.621
SmolLM2-135M-Instruct	1	8	4	1.0e-4	generation	0.562
SmolLM2-135M	2	2	4	1.0e-4	head	0.473

Table 4: Hyperparameter sweep with $F1_{macro}$ across all datasets.

Hyperparameter	Tested values
LR.	[0.001, 0.01, 0.1, 0.2, 0.5, 1.0]
NGD LR.	[0.05, 0.1, 0.2, 0.5]
Num. Data Points	[20, 100, 200]
σ_{noise}	[0.2, 0.5]
σ_{walk}	[0.1, 0.5]

Table 5: NGD LR. stands for natural gradient descent optimizer learning rate. It is typical to use a large learning rate for the learning rate of natural gradient descent (Salimbeni et al., 2018).

We then use the observed classifications $\hat{s}$ and observed timestamps t_{obs} as the observations, and the full timestamp vector t as the timestamps to infer the latent stance on.

C Learning Rate Hyperparameter Sweep

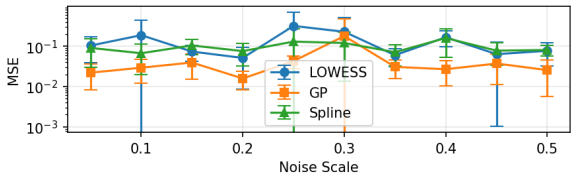
We trialled several hyperparameter settings when searching for the best learning rates and learning rate schedulers. We detail them in Table 5, alongside logging the number of epochs needed to achieve 90% loss reduction as a measure of convergence speed.

We report the aggregated scheduler metrics in Table 6.

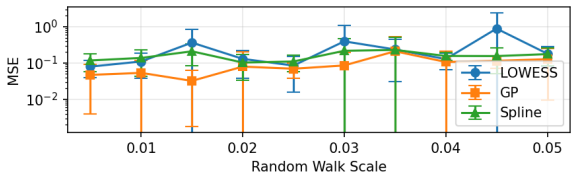
D Time Series Inference Results

Synthetic data generation parameter sensitivity experiment outputs are shown in Figs. 5a and 5b.

Overall output metrics from our time-series inference comparison experiments are shown in Table 7.



(a) Adjusting noise scale from 0.05 to 0.5.



(b) Adjusting random walk scale from 0.005 to 0.05.

Figure 5: Mean MSE from 5 runs for each of 10 values of an adjusted synthetic data generation parameter, used to compare the GP, LOWESS, and spline models.

Scheduler	Min. Loss	Num Epochs Conv.
None	1.12	97
Cosine Warm Restarts	1.19	112
Cosine	1.19	73
Step	1.22	74
Exponential	1.22	87

Table 6: Evaluation of learning rate schedulers, using minimum loss achieved and number of epochs to 90% loss reduction as measures of efficacy and speed, respectively.

Method	MSE	Training Time
GP	0.069 ± 0.13	21.7 ± 7.2
LOWESS	0.197 ± 0.46	0.003 ± 0.001
Spline	0.130 ± 0.12	0.002 ± 0.007

Table 7: Comparison of time series inference methods.
Numbers listed are mean $\pm$ standard deviation.