BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan mengenai analisis sentimen masyarakat terhadap Program Makan Bergizi Gratis (MBG) pada media sosial TikTok menggunakan kombinasi model IndoBERT dan *Support Vector Machine* (SVM), maka dapat diperoleh beberapa kesimpulan sebagai berikut:

1. Distribusi sentimen masyarakat didominasi oleh sentimen negatif sebanyak 2.106 komentar, diikuti sentimen positif sebanyak 745 komentar, dan sentimen netral sebanyak 739 komentar dari total 3.590 komentar. Hasil tersebut menunjukkan bahwa opini masyarakat terhadap implementasi Program Makan Bergizi Gratis masih didominasi oleh berbagai tanggapan kritis, meskipun sebagian masyarakat juga memberikan dukungan terhadap tujuan dan manfaat program.
2. Penerapan metode IndoBERT–*Support Vector Machine* (SVM) mampu mengklasifikasikan sentimen komentar TikTok dengan performa yang baik. Berdasarkan empat skenario pengujian, performa terbaik diperoleh pada dataset *non-hybrid* dengan penerapan *class weight* yang menghasilkan akurasi sebesar 83,70%, *precision* sebesar 84,43%, *recall* sebesar 77,74%, dan F1-*score* sebesar 80,43%. Hasil tersebut

menunjukkan bahwa representasi fitur menggunakan *embedding* IndoBERT yang dikombinasikan dengan algoritma SVM efektif dalam melakukan klasifikasi sentimen pada data komentar berbahasa Indonesia yang bersifat informal.

3. Pengaruh distribusi data terhadap kinerja model menunjukkan bahwa penerapan *class weight* pada dataset yang masih mengalami ketidakseimbangan kelas mampu meningkatkan performa model dibandingkan tanpa pembobotan. Sementara itu, proses *hybrid labelling* tidak menghasilkan peningkatan performa klasifikasi dibandingkan dataset *non-hybrid*. Namun demikian, *hybrid labelling* berhasil meningkatkan kualitas *ground truth* melalui validasi manual terhadap data ambigu oleh tiga annotator berdasarkan pedoman anotasi, yang didukung oleh hasil *Intra-Annotator Consistency Check* dengan nilai *Cohen's Kappa* sebesar 0,9197 (*Almost Perfect Agreement*). Dengan demikian, penelitian ini menunjukkan bahwa *class weight* lebih berpengaruh terhadap peningkatan performa model, sedangkan *hybrid labelling* berkontribusi dalam meningkatkan validitas dan reliabilitas label pada dataset.

5.2 Saran

Berdasarkan hasil penelitian yang telah dilakukan, beberapa saran yang dapat diberikan untuk penelitian selanjutnya maupun pihak terkait adalah sebagai berikut.

1. Menggunakan jumlah data yang lebih besar dan berasal dari berbagai platform media sosial, seperti X (*Twitter*), *Instagram*, *Facebook*, *YouTube*, maupun portal berita daring sehingga dapat memberikan gambaran sentimen masyarakat yang lebih luas dan representatif.
2. Membandingkan performa *Support Vector Machine* (SVM) dengan algoritma klasifikasi lainnya, seperti *Random Forest*, XGBoost, *Logistic Regression*, maupun model berbasis *deep learning* untuk mengetahui metode yang paling optimal dalam melakukan klasifikasi sentimen.
3. Menerapkan metode penanganan ketidakseimbangan data lainnya, seperti SMOTE, ADASYN, atau teknik *oversampling* dan *undersampling* untuk membandingkan efektivitasnya dengan pendekatan *class weight*.
4. Mengembangkan proses *hybrid labelling* dengan melibatkan lebih banyak annotator serta melakukan pengujian *Inter-Annotator Agreement* selain *Intra-Annotator Consistency*, sehingga kualitas *ground truth* yang dihasilkan dapat diukur secara lebih komprehensif.
5. Bagi pemerintah dan pemangku kebijakan, hasil penelitian ini diharapkan dapat menjadi bahan evaluasi dalam meningkatkan kualitas implementasi Program Makan Bergizi Gratis, khususnya terhadap isu-isu yang banyak menjadi perhatian masyarakat, seperti kualitas makanan, keamanan pangan, transparansi pelaksanaan, dan efektivitas penggunaan anggaran, sehingga tingkat kepercayaan dan penerimaan masyarakat terhadap program dapat semakin meningkat.

DAFTAR PUSTAKA

Adhim, N. F., & Cahyono, N. (2025). Optimization of IndoBERT for Sentiment Analysis of FOMO on Social Media Through Fine-Tuning and Hybrid Labeling. *Journal of Applied Informatics and Computing (JAIC)*, *9*(6), 3786–3797.

Akbar, H., & Sanjaya, W. K. (2023). Kajian Performa Metode Class Weight Random Forest pada Klasifikasi Imbalance Data Kelas Curah Hujan. *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, *3*(1). https://doi.org/10.20885/snati.v3i1.30

Alfataah, S., & Cahyono, N. (2025). Sentimen Analisis Pengguna Twitter Terhadap Provider Xl Axiata Menggunakan Metode Support Vector Machine. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *9*(3), 4502–4506. https://doi.org/10.36040/jati.v9i3.13647

Al-Kadzim, M. G., Rasim, R., & Herbert, H. (2025). Analisis Perubahan Sentimen Publik di Media Sosial X terhadap Konflik Palestina-Israel Menggunakan Model IndoBERT. *Digital Transformation Technology*, *4*(2), 1167–1174. https://doi.org/10.47709/digitech.v4i2.5312

Amarulloh, A., Kurniasih, K., & Muchlis, M. (2023). Analisis Perbandingan Performa Web Service Rest Menggunakan Framework Laravel, Django, dan Node JS Untuk Akses Dsta Dengan Aplikasi Website. *Jurnal Teknik Informatika STMIK Antar Bangsa*, *9*(1), 14–19. https://doi.org/10.51998/jti.v9i1.515

Aprillia, N. P., & Azzahra, M. U. D. (2025). Program Makan Bergizi Gratis (MBG): Antara Legacy Project dan Solusi Stunting. *Journal of Information Systems and Management (JISMA)*, *4*(5), 19–22. https://doi.org/10.4444/jisma.v4i5.1229

Arif, M. W., & Kustiyono, K. (2025). Analisis Sentimen Kebijakan Makan Bergizi Gratis di Media Sosial Menggunakan Natural Language Processing Berbasis Python TextBlob di Indonesia. *Jurnal Pendidikan dan Teknologi Indonesia (JPTI)*, *5*(9), 2463–2471. https://doi.org/10.52436/1.jpti.931

Asshiddiq, Muh. H., & Witanti, A. (2025). Analisis Sentimen tentang Penundaan Pengangkatan CPNS 2025 pada Platform X Menggunakan Metode IndoBERT. *Jurnal teknika*, *17*(2), 97–108. https://doi.org/10.30736/jt.v17i2.1448

Badan Gizi Nasional. (2025). *BGN Tanggapi Serius Dugaan Insiden Keracunan Pangan Program MBG di Sekolahil*. Badan Gizi Nasional. https://bgn.go.id/news/siaran-pers/bgn-tanggapi-serius-dugaan-insiden-keracunan-pangan-program-mbg-di-sekolah

Bader, M., Abdelwanis, M., Maalouf, M., & Jelinek, H. F. (2024). Detecting depression severity using weighted random forest and oxidative stress

biomarkers. *Scientific Reports*, *14*(1), 16328. https://doi.org/10.1038/s41598-024-67251-y

Basyari, I. (2025, January 3). *Program Makan Bergizi Gratis Dimulai Senin Besok, 6 Januari*. Kompas.id. https://www.kompas.id/artikel/program-makan-bergizi-gratis-dimulai-6-januari

Berutu, S. S., Budiati, H., Jatmika, J., & Gulo, F. (2023). Data preprocessing approach for machine learning-based sentiment classification. *JURNAL INFOTEL*, *15*(4), 317–325. https://doi.org/10.20895/infotel.v15i4.1030

Chamid, A. A., Widowati, & Kusumaningrum, R. (2024). Labeling Consistency Test of Multi-Label Data for Aspect and Sentiment Classification Using the Cohen Kappa Method. *Ingénierie Des Systèmes d Information*, *29*(1), 161–167. https://doi.org/10.18280/isi.290118

Darmawan, Z. M. E., & Dianta, A. F. (2023). Implementasi Optimasi Hyperparameter GridSearchCV Pada Sistem Prediksi Serangan Jantung Menggunakan SVM. *Teknologi: Jurnal Ilmiah Sistem Informasi*, *13*(1), 8–15. https://doi.org/10.26594/teknologi.v13i1.3098

Dasriani, N. G. A., Pariandi, L. A. G., & Dharma, I. M. Y. (2024). Analisis Sentimen Program Jaminan Kesehatan Nasional Menggunakan Multiclass Support Vector Machine. *BIOS : Jurnal Teknologi Informasi dan Rekayasa Komputer*, *6*(1), 20–30. https://doi.org/10.37148/bios.v6i1.136

Destryanto, F., Rizqiyah, P., & Sokibi, P. (2025). Analisis Sentimen Respon Publik terhadap Program Makan Bergizi Gratis di Instagram Menggunakan IndoBERT. *Jurnal Kecerdasan Buatan dan Aplikasi Rekayasa*, *5*(1).

Dewi, F., Wibowo, N. C. H., Handayani, M. R., & Umam, K. (2025). Evaluasi Hyperparameter Tuning Pada Support Vector Machine (SVM) Dalam Klasifikasi Ulasan Hotel Di Tripadvisor. *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, *10*(3), 2584–2593. https://doi.org/10.29100/jipi.v10i3.7774

Dewi, M. S., & Hasugian, A. H. (2025). Penerapan Supprot Vector Machine Untuk Analisis Sentimen Pada Tanggapan Masyarakat di Media Sosial Terhadap Program Makan Bergizi Gratis: Application of Support Vector Machine for Sentiment Analysis on Publik Response Towards Free Lunch Program. *Jurnal Teknologi Dan Sistem Informasi Univrab (RABIT)*, *10*(2), 911–922.

Fadilah, G. T., Muflikhah, L., & Perdana, R. S. (2025). Analisis Sentimen Produk Hijab Pada E-commerce Tokopedia Menggunakan Algoritma Support Vector Machine dan IndoBERT Embedding. *J-PTIIK (Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer)*, *9*(2).

Fahreza, M. H., Sipayung, N. Z., Santoso, I. W., & Susilowati, S. (2026). *Komparasi SVM-LR Berbasis IndoBERT dan SMOTE-TOMEK Pada Deteksi Buzzer*. *7*(2).

Febry, A., & Safitri, D. (2021). Dismilaritas Kecanduan Pemakaian Media Sosial Generasi Y dan Generasi Z. *Edukasi IPS*, *5*(2), 37–45. https://doi.org/10.21009/EIPS.005.02.05

Firdaus, R., Asror, I., & Herdiani, A. (2021). Lexicon-Based Sentiment Analysis of Indonesian Language Student Feedback Evaluation. *Indonesian Journal on Computing (Indo-JC)*, 1-12 Pages. https://doi.org/10.34818/INDOJC.2021.6.1.408

Hamka, M., & Ratna Sari, D. (2022). Analisis Sentimen Dan Information Extraction Pembelajaran Daring Menggunakan Pendekatan Lexicon. *Djtechno: Jurnal Teknologi Informasi*, *3*(1), 21–32. https://doi.org/10.46576/djtechno.v3i1.2194

Handayani, A., & Zufria, I. (2023). *Analisis Sentimen Terhadap Bakal Capres RI 2024 di Twitter Menggunakan Algoritma SVM*. *5*(1), 53–56.

Hidayatulloh, S., Muflikhah, L., & Perdana, R. S. (2025). Implementasi Embedding IndoBERT dan Support Vector Machine (SVM) Untuk Analisis Sentimen Publik terhadap Layanan Biznet. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (JPTIIK)*, *9*(9).

Hussain, N., Qasim, A., Mehak, G., Zain, M., Sidorov, G., Gelbukh, A., & Kolesnikova, O. (2025). Multi-Level Depression Severity Detection with Deep Transformers and Enhanced Machine Learning Techniques. *AI*, *6*(7), 157. https://doi.org/10.3390/ai6070157

Ibrahim, R. A., Amri, D., & Kamal. (2025, September 22). *MBG dan ribuan kasus keracunan – Evaluasi SPPG dan standar higienis jadi prioritas*. BBC News Indonesia. https://www.bbc.com/indonesia/articles/cgj119359q4o

Ihalauw, S. A., Trezandy Lapatta, N., Wiria Nugraha, D., Wirdayanti, & Ar Lamasitudju, C. (2025). Analisis Sentimen Terhadap Kinerja Awal Pemerintahan Menggunakan IndoBERT Dan SMOTE Pada Media Sosial X. *Jurnal Algoritma*, *22*(2). https://doi.org/10.33364/algoritma/v.22-2.2957

Imaduddin, H., A'la, F. Y., & Nugroho, Y. S. (2023). Sentiment Analysis in Indonesian Healthcare Applications using IndoBERT Approach. *International Journal of Advanced Computer Science and Applications(IJACSA)*, *14*(8), 113–117. https://doi.org/https://doi.org/10.14569/IJACSA.2023.0140813

Imron, S., Setiawan, E. I., & Santoso, J. (2023). Deteksi Aspek Review E-Commerce Menggunakan IndoBERT Embedding dan CNN. *Journal of Intelligent System and Computation*, *5*(1), 10–16. https://doi.org/10.52985/insyst.v5i1.267

Isnur, P. (2025). *Makan Gratis, Ekonomi Tumbuh*. Indonesia Baik.Id. http://indonesiabaik.id/infografis/makan-gratis-ekonomi-tumbuh?utm_source=chatgpt.com

Jhosefhin, N. V. R., & Dewi, C. (2025). Analisis Sentimen Crawling Data dari Sosial Media X tentang Gaza Menggunakan Metode SVM dan Decision Tree. *Jurnal Indonesia : Manajemen Informatika Dan Komunikasi*, *6*(1), 427–437. https://doi.org/10.35870/jimik.v6i1.1225

Kasparov, M. C. B., Darmawan, I., & Pratiwi, O. N. (2024). Rancang Bangun Website Fund Myedu Untuk Mencari Beasiswa Dengan Menggunakan Crawling Api. *eProceedings of Engineering*, *11*(4), 4319–4326.

Londjo, M. F. (2021). Implementasi White Box Testing Dengan Teknik Basis Path Pada Pengujian Form Login. *Jurnal Siliwangi*, *7*(2), 35–40.

Manoppo, M. R., Kolang, I. C., Fiat, D. N. N., Mawara, R. M. C., Sumarno, A. D. P., Yusupa, A., & Tarigan, V. (2025). Analisis Sentimen Publik Di Media Sosial Terhadap Kenaikan PPN 12% Di Indonesia Menggunakan IndoBERT. *Jurnal Kecerdasan Buatan Dan Teknologi Informasi*, *4*(2), 152–163. https://doi.org/10.69916/jkbti.v4i2.322

Mauliza, A., & Abdi, M. (2025). *Cyberbullying Detection Model Using IndoBERT Representation and Support Vector Machine*. *1*(2), 187–199.

Melinda, A. D., & Sari, V. (2026). Analisis Sentimen Ulasan Pengguna Aplikasi Line Bank Di Google Play Store Menggunakan Metode Support Vector Machine Berbasis Smote Dengan Optimasi Gridsearchcv. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *10*(3), 5151–5158. https://doi.org/10.36040/jati.v10i3.18337

Musfiroh, D., Khaira, U., Utomo, P. E. P., & Suratno, T. (2021). Analisis Sentimen terhadap Perkuliahan Daring di Indonesia dari Twitter Dataset Menggunakan InSet Lexicon: Sentiment Analysis of Online Lectures in Indonesia from Twitter Dataset Using InSet Lexicon. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *1*(1), 24–33. https://doi.org/10.57152/malcom.v1i1.20

Mustofa, M. B., Prastya, I. W. D., & Sahri, S. (2026). Analisis Sentimen Multi-Platform Media Sosial pada Program Makan Bergizi Gratis Menggunakan Ensemble IndoBERT-SVM. *JURIKOM (Jurnal Riset Komputer)*, *13*(1), 265–264. https://doi.org/10.30865/jurikom.v13i1.9460

Naibaho, D. O., Setiawan, N. Y., & Nugraha, D. C. A. (2026). Analisis Komparatif Sentimen Berbasis Aspek Pada Ulasan Konsumen Café: Studi Perbandingan Model Bahasa Gemma Dan Indobert-Svm. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (JPTIIK)*, *9*(12).

Nasrullah, N., & Bachtiar, Muh. Y. (2021). Inovasi Pembelajaran Daring dan Dampak Bagi PAUD Selama Pandemi Covid-19. *Jurnal Obsesi : Jurnal Pendidikan Anak Usia Dini*, *6*(2), 1007–1019. https://doi.org/10.31004/obsesi.v6i2.1411

Nazar, R. (2024). Implementasi Pemrograman Python Menggunakan Google Colab. *Jurnal Informatika dan Komputer (JIK)*, *15*(1), 50–56.

Ningsih, W., Alfianda, B., Rahmaddeni, R., & Wulandari, D. (2024). Perbandingan Algoritma SVM dan Naïve Bayes dalam Analisis Sentimen Twitter pada Penggunaan Mobil Listrik di Indonesia. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, *4*(2), 556–562. https://doi.org/10.57152/malcom.v4i2.1253

Nurhidayat, R., & Dewi, K. E. (2023). Penerapan Algoritma K-Nearest Neighbor Dan Fitur Ekstraksi n-Gram Dalam Analisis Sentimen Berbasis Aspek. *Komputa : Jurnal Ilmiah Komputer dan Informatika*, *12*(1), 91–100. https://doi.org/10.34010/komputa.v12i1.9458

Omar, A., & Abd El-Hafeez, T. (2024). Optimizing epileptic seizure recognition performance with feature scaling and dropout layers. *Neural Computing and Applications*, *36*(6), 2835–2852. https://doi.org/10.1007/s00521-023-09204-6

Pradnyana, G. A., Anggraeni, W., Yuniarno, E. M., & Purnomo, M. H. (2023). Fine-Tuning IndoBERT Model for Big Five Personality Prediction from Indonesian Social Media. *International Seminar on Intelligent Technology and Its Applications (ISITIA)*, 93–98. https://doi.org/https://doi.org/10.46229/jifotech.v5i1.978

Pranata, J., Agustian, S., & Elin Haerani, J. (2024). Penggunaan Model Bahasa indoBERT pada Metode Random Forest Untuk Klasifikasi Sentimen Dengan Dataset Terbatas. *Building of Informatics, Technology and Science (BITS)*, *6*(3). https://doi.org/https://doi.org/10.47065/bits.v6i3.6335

Prastia, A. L., & Asra, T. (2026). Perbandingan Algoritma Random Forest, SVM, Dan Naive Bayes Dalam Analisis Sentimen Ulasan Spotify Di Play Stire Berbasis SMOTE. *Jurnal Informatika Kaputama (JIK)*, *10*(1).

Purwowidhu. (2025). *Menilik Eksistensi Makan Bergizi Gratis alias MBG*. Media Keuangan. https://mediakeuangan.kemenkeu.go.id/article/show/menilik-eksistensi-program-mbg-atau-makan-bergizi-gratis

Putriyekti, A., Sulistiawati, D., Khairani, N., & Kusuma, R. R. A. (2025). Analisis Multidimensional Sentimen Masyarakat terhadap Program Makan Bergizi Gratis pada Media Sosial X. *Integrative Perspectives of Social and Science Journal (IPSSJ)*, *2*(1), 1004–1024.

Qolbu, A. A., Fitriyati, N., & Inayah, N. (2025). Performa Naïve Bayes, SVM, dan IndoBERT pada Analisis Sentimen Twitter IndiHome dengan Strategi Penanganan Data Tidak Seimbang. *Jurnal Fourier*, *14*(1), 29–44. https://doi.org/10.14421/fourier.2025.141.29-44

Report. (2024). *Efek Pengganda Program Makan Bergizi Gratis*. Institute for Development of Economics and Finance (INDEF).

Ritonga, I. S., Wanayumini, & Hartama, D. (2025). Sentiment Classification in Imbalanced Data: Trade-Offs Between Metrics and Real-World Relevance. *JURNAL TEKNIK INFORMATIKA*, *18*(2), 303–315. https://doi.org/10.15408/jti.v18i2.46652

Riyanto, A. D. (2022, February 18). Hootsuite (We are Social): Indonesian Digital Report 2022. *Dosen, Praktisi, Konsultan, Pembicara/Fasilitator Digital Marketing, Internet marketing, SEO, Technopreneur dan Bisnis Digital*. https://andi.link/hootsuite-we-are-social-indonesian-digital-report-2022/

Rolangon, A., Weku, A., & Sandag, G. A. (2023). Perbandingan Algoritma LSTM Untuk Analisis Sentimen Pengguna Twitter Terhadap Layanan Rumah Sakit Saat Pandemi Covid-19. *TeIKa*, *13*(1), 31–40. https://doi.org/10.36342/teika.v13i01.3063

Rozi, I. F., Maulidia, I., Hani'ah, M., Arianto, R., Yunianto, D. R., & Ananta, A. Y. (2025). Comparison of Feature Extraction in Support Vector Machine (SVM) Based Sentiment Analysis System. *Jurnal Ilmiah Kursor*, *13*(1), 1–12. https://doi.org/10.21107/kursor.v13i1.417

Rustandi, S. R. K. W. T., Suhaedi, D., & Pemanasari, Y. (2023). Pemetaan Hyperplane Pada Support Vector Machine. *Bandung Conference Series: Mathematics*, *3*(2), 109–119. https://doi.org/10.29313/bcsm.v3i2.8187

Saifullah, S., Drężewski, R., Dwiyanto, F. A., Aribowo, A. S., Fauziah, Y., & Cahyana, N. H. (2024). Automated Text Annotation Using a Semi-Supervised Approach with Meta Vectorizer and Machine Learning Algorithms for Hate Speech Detection. *Applied Sciences*, *14*(3), 1078. https://doi.org/10.3390/app14031078

Saleh, M. F., & Imanda, R. (2025). Public Sentiment Analysis of the Free Meal Program: A Comparison of Naive Bayes and Support Vector Machine Methods on the Twitter (X) Social Media Platform. *Journal of Applied Informatics and Computing*, *9*(1), 131–139. https://doi.org/10.30871/jaic.v9i1.8895

Suadaa, L. H., Santoso, I., & Panjaitan, A. T. B. (2021). Transfer Learning of Pre-trained Transformers for Covid-19 Hoax Detection in Indonesian Language. *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, *15*(3), 317. https://doi.org/10.22146/ijccs.66205

Sulistyo, D. A., & Setiadi, E. (2025). Analisis Sentimen Kebijakan Makan Bergizi Gratis Menggunakan IndoBERT dan Machine Learning. *JURNAL FASILKOM*, *15*(3), 507–513. https://doi.org/10.37859/jf.v15i3.10546

Sunarko, P., Negara, A. B. P., & Septiriana, R. (2024). Perbandingan Klasifikasi Algoritma Support vector machine dan Naïve Bayes Menggunakan Labeling VADER dan Lexicon based pada Tweets Bahasa Indonesia dan Bahasa Inggris. *Jurnal aplikasi dan Riset Informatika (JUARA)*, *3*(1), 9–19.

Sutisna, A. C., Pramudita, A. E., Syahputra, I. Y. A., & Rakhmawati, N. A. (2025). Analisis Sentimen Penggunaan Platform Cloud Storage Pada Mahasiswa Dengan Metode Naive Bayes (studi Kasus Pada Mahasiswa Sarjana Sistem Informasi Its). *Etika Teknologi Informasi*, *1*(1).

Tandiwijaya, R., & Darmawan, J. B. B. (2026). Perbandingan Algoritma Gaussian Naïve Bayes dan K-Nearest Neighbor dalam Klasifikasi Dataset Employee

Turnover. *Prosiding Seminar Nasional Informatika Bela Negara (SANTIKA)*, *6*(1), 224–249. https://doi.org/10.33005/santika.v6i1.1288

Toha, A., Purwono, P., & Gata, W. (2022). Model Prediksi Kualitas Udara dengan Support Vector Machines dengan Optimasi Hyperparameter GridSearch CV. *Buletin Ilmiah Sarjana Teknik Elektro*, *4*(1), 12–21. https://doi.org/10.12928/biste.v4i1.6079

Triningsih, E., Afdal, M., Permana, I., & Rozanda, N. E. (2025). Analisis Sentimen Terhadap Program Makan Bergizi Gratis Menggunakan Algoritma Machine Learning Pada Sosial Media X. *Building of Informatics, Technology and Science (BITS)*, *6*(4), 2240–2250. https://doi.org/10.47065/bits.v6i4.6534

Utary, S., Jannah, R. Z., Asmawita, P., & Pujianto. (2026). 2335-9675 Klasifikasi Sentimen Komentar TikTok Terkait Kenaikan Harga BBM 2026 di Sumatera Selatan Menggunakan Algoritma Naive Bayes pada Orange Data Mining. *JSI (Jurnal Sistem Informasi) Universitas Suryadarma*, *13*(2), 298–305. https://doi.org/10.35968/jsi.v13i2.2108

Vallen, J., & Azizah, A. H. (2025). Analisis Sentimen Media Sosial X Program Makanan Sehat Gratis dengan Support Vector Machine dan Naive Bayes. *Jurnal ilmiah Sistem Informasi dan Ilmu Komputer*, *5*(3), 308–318. https://doi.org/10.55606/juisik.v5i3.1665

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762

Wahyudi, M. F. D., Chrisvo, A. R., Hidayatullah, M. F., & Parhusip, J. (2024). Analisis Distribusi Selisih Tingkat Pengangguran Terbuka (TPT) Dan Tingkat Partisipasi Angkatan Kerja (TPAK) Menggunakan Google Colab. *Informatika: Jurnal Teknik Informatika Dan Multimedia*, *4*(2), 39–45. https://doi.org/10.51903/informatika.v4i2.828

Wibowo, J. A., Mawardi, V. C., & Sutrisno, T. (2024). Visualisasi Word Cloud Hasil Analisis Sentimen Berbasis Fitur Layanan Aplikasi Gojek dengan Support Vector Machine. *Jurnal Serina Sains, Teknik dan Kedokteran*, *2*(1), 61–70. https://doi.org/10.24912/jsstk.v2i1.32058

LAMPIRAN

Lampiran 1

Kode Program

```
from google.colab import drive

drive.mount('/content/drive')

#PROSES CLEANING DATA

import pandas as pd

import re

df = pd.read_csv('/content/drive/MyDrive/SKRIPSI/SKRIPSI2/dataset_tiktok-clean.csv')

def cleaning(text):
    text = str(text)
    # hapus URL
    text = re.sub(r'http\S+|www\S+|https\S+', '', text)
    # hapus mention
    text = re.sub(r'@\w+', '', text)
    # hapus hashtag
    text = re.sub(r'#\w+', '', text)
    # hapus simbol dan emoji
    # angka tetap dipertahankan
    text = re.sub(r'[^a-zA-Z0-9\s]', '', text)
    return text

df['CLEANING'] = df['TEKS'].apply(cleaning)

tabel_cleaning = df[['TEKS', 'CLEANING']]

tabel_cleaning.head(10)

tabel_cleaning.to_csv('/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_cleaning.csv', index=False)

print("Jumlah data sebelum hapus duplikat:", len(df))
```

```
#HANDLING DUPLICATE
df = df.drop_duplicates(subset=['CLEANING'])
df = df.reset_index(drop=True)
print("Jumlah data setelah hapus duplikat:", len(df))
tabel_duplicate.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_handling_duplicate.csv',
    index=False)
#PROSES CASE FOLDING
def case_folding(text):
    text = text.lower()
    return text
df['CASE_FOLDING'] = df['CLEANING'].apply(case_folding)
tabel_casefolding = df[['CLEANING', 'CASE_FOLDING']]
tabel_casefolding.head(10)
tabel_casefolding.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_casefolding.csv',
    index=False)
#PROSES NORMALISASI
slang_df = pd.read_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/slang_indo.csv',
    sep=';')
slang_dict = dict(zip(slang_df['tidak baku'], slang_df['baku']))
def normalisasi(text):
    kata = text.split()
    hasil = []
    for i in kata:
        if i in slang_dict:
```

```
            hasil.append(slang_dict[i])
        else:
            hasil.append(i)
    return ' '.join(hasil)
df['NORMALISASI'] = df['CASE_FOLDING'].apply(normalisasi)
tabel_normalisasi = df[['CASE_FOLDING', 'NORMALISASI']]
tabel_normalisasi.head(10)
tabel_normalisasi.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_normalisasi.csv',
    index=False)
#HYBRID LABELLING
#1. Pre-Trained labelling IndoBERT
!pip install transformers
!pip install torch
from transformers import pipeline
import pandas as pd
classifier = pipeline(
    "sentiment-analysis",
    model="mdhugol/indonesia-bert-sentiment-classification")
label_map = {
    'LABEL_0': 'positif',
    'LABEL_1': 'netral',
    'LABEL_2': 'negatif'}
#auto-labelling-non-hybrid
hasil_label = []
for text in df['NORMALISASI']:
    result = classifier(str(text))[0]
```

```
    label_asli = result['label']
    label_sentimen = label_map[label_asli]
    hasil_label.append({
        'NORMALISASI': text,
        'LABEL': label_sentimen,
        'CONF_SCORE': result['score'] })
classifier.model.config.id2label
label_df = pd.DataFrame(hasil_label)
label_df.head(10)
jumlah_sentimen = label_df['LABEL'].value_counts().reset_index()
jumlah_sentimen.columns = ['Sentimen', 'Jumlah']
jumlah_sentimen
# distribusi diagram auto-labelling
import matplotlib.pyplot as plt
urutan = [
    'positif',
    'netral',
    'negatif']
# hitung jumlah label dan urutkan
label_count = label_df[
    'LABEL'
].value_counts().reindex(
    urutan,
    fill_value=0)
# warna tiap sentimen
color_map = {
    'positif': 'green',
```

```
    'netral': 'gray',
    'negatif': 'red'}
colors = [
    color_map[label]
    for label in label_count.index]
# buat diagram batang
ax = label_count.plot(
    kind='bar',
    figsize=(6,4),
    color=colors)
# judul dan label
plt.title(
    'Distribusi Hasil Auto Labeling IndoBERT')
plt.xlabel('Sentimen')
plt.ylabel('Jumlah Data')
plt.xticks(rotation=0)
# tampilkan jumlah di atas batang
for i in ax.patches:
    ax.annotate(
        str(int(i.get_height())),
        (
            i.get_x() + i.get_width()/2,
            i.get_height()
        ),
        ha='center',
        va='bottom')
plt.show()
```

```
label_df_csv = label_df.copy()
label_df_csv['CONF_SCORE'] = label_df_csv['CONF_SCORE'].round(4)
label_df_csv.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_auto_labeling_indobert.csv',
    index=False,
    encoding='utf-8-sig')
#menentukan Threshold dengan persenti 25%
import matplotlib.pyplot as plt
threshold = label_df['CONF_SCORE'].quantile(0.20)
print("Threshold:", threshold)
#pisahkan data
#confident data
data_confident = label_df[
    label_df['CONF_SCORE'] >= threshold]
#low confident data
data_low = label_df[
    label_df['CONF_SCORE'] < threshold]
# menentukan persentil
persentil = 0.20
# menghitung threshold
threshold = label_df['CONF_SCORE'].quantile(persentil)
print("Threshold :", threshold)
print("Persentil :", persentil * 100, "%")
# membuat histogram
plt.hist(label_df['CONF_SCORE'], bins=30)
# garis threshold
plt.axvline(
```

```
    threshold,
    linestyle='dashed',
    label=f'Threshold = {threshold:.3f}\nPersentil = {persentil*100:.0f}%')
# label grafik
plt.xlabel('Confidence Score')
plt.ylabel('Jumlah Data')
plt.title('Distribusi Confidence Score')
# tampilkan legenda
plt.legend()
plt.show()
# 2. VALIDASI LEXICON berdasarkan threshold
#kamus inset lexicon posotif dan negatif
positive = pd.read_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/positive.tsv',
    sep='\t')
negative = pd.read_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/negative.tsv',
    sep='\t')
print("Jumlah kata positif :", len(positive))
print("Jumlah kata negatif :", len(negative))
total = len(positive) + len(negative)
print("Total kamus leksikon :", total)
#ubah le dictionary
positive_dict = dict(
    zip(
        positive['word'],
        positive['weight']
```

```
    ))
negative_dict = dict(
    zip(
        negative['word'],
        negative['weight']))
#fungsi labeling leksikon
def lexicon_label(text):
    score = 0
    words = text.split()
    for word in words:
        if word in positive_dict:
            score += positive_dict[word]
        elif word in negative_dict:
            score += negative_dict[word]
    if score > 0:
        return 'positif'
    elif score < 0:
        return 'negatif'
    else:
        return 'netral'
#terapkan pada data
data_low['LEXICON_LABEL'] = data_low[
    'NORMALISASI'
].apply(lexicon_label)
#data valid
data_valid_low = data_low[
    (data_low['LABEL'] ==
```

```
    data_low['LEXICON_LABEL'])]
#data ambigu
data_ambigu = data_low[
    ((data_low['LABEL'] == 'positif') &
        (data_low['LEXICON_LABEL'] == 'negatif'))
    |
    ((data_low['LABEL'] == 'negatif') &
        (data_low['LEXICON_LABEL'] == 'positif'))
    |
    ((data_low['LABEL'] == 'netral') &
        (data_low['LEXICON_LABEL'] == 'positif'))
    |
    ((data_low['LABEL'] == 'netral') &
        (data_low['LEXICON_LABEL'] == 'negatif'))
    |
    ((data_low['LABEL'] == 'positif') &
        (data_low['LEXICON_LABEL'] == 'netral'))
    |
    ((data_low['LABEL'] == 'negatif') &
        (data_low['LEXICON_LABEL'] == 'netral'))]
data_valid_final = pd.concat([
    data_confident,
    data_valid_low])
print("Jumlah Data valid confident :", len(data_confident))
print("Jumlah Data valid low :", len(data_valid_low))
print("Jumlah Data Valid :", len(data_valid_final))
print("Jumlah Data Ambigu :", len(data_ambigu))
```

```
#simpan data valid
data_valid_final.to_csv(
   '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_valid_lexicon1.csv',
   index=False)
#simpan data ambigu
data_ambigu.to_csv(
   '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_ambigu1.csv',
   index=False)
data_ambiguauto = pd.read_csv(
   '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_ambigu1.csv')
# 3. CEK MANUAL dan Intra-Annotator Consistency Check
from sklearn.metrics import cohen_kappa_score
data_ambigu = pd.read_csv(
   '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_ambigu_manual1.csv')
ambiguauto_count = data_ambiguauto['LABEL'].value_counts()
print(ambiguauto_count)
#buat distibusi data ambigu yang dilabeli manual
ambigu_count = data_ambigu['FINAL'].value_counts()
print(ambigu_count)
jumlah_sample = int(len(data_ambigu) * 0.10)
print("Jumlah Sampel :", jumlah_sample)
#Intra-Annotator Consistency Check
#ambil sampel acak
sample_check = data_ambigu.sample(
   n=jumlah_sample,
   random_state=42)
label_asli = sample_check[
```

```
    ['NORMALISASI', 'P1']]
#buat file recheck
recheck_file = sample_check[
    ['NORMALISASI']
].copy()
recheck_file['RECHECK_LABEL'] = ''
recheck_file.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/sample_recheck.csv',
    index=False)
hasil_recheck = pd.read_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_sample_recheck1.csv')
compare_df = pd.merge(
    label_asli,
    hasil_recheck,
    on='NORMALISASI')
from sklearn.metrics import cohen_kappa_score
kappa = cohen_kappa_score(
    compare_df['P1'],
    compare_df['RECHECK_LABEL'])
print(
    "Cohen's Kappa :",
    round(kappa, 4))
jumlah_sama = (
    compare_df['P1']
    ==
    compare_df['RECHECK_LABEL']
).sum()
```

```
persentase = (
    jumlah_sama /
    len(compare_df)
) * 100
print(
    "Persentase Kesesuaian :",
    round(persentase, 2),
    "%")
if kappa >= 0.81:
    interpretasi = "Perfect Agrimeent"
elif kappa >= 0.61:
    interpretasi = "Near Perfect Agreement"
elif kappa >= 0.41:
    interpretasi = "Moderate Agreement"
elif kappa >= 0.21:
    interpretasi = "Fair Agreement"
else:
    interpretasi = "Slight Agreement"
print("Interpretasi :", interpretasi)
#confusion matrix intra annotator
from sklearn.metrics import (
    confusion_matrix,
    ConfusionMatrixDisplay)
import matplotlib.pyplot as plt
cm = confusion_matrix(
    compare_df['P1'],
    compare_df['RECHECK_LABEL'],
```

```
    labels=[
        'negatif',
        'netral',
        'psoitif'])
disp = ConfusionMatrixDisplay(
    confusion_matrix=cm,
    display_labels=[
        'negatif',
        'netral',
        'positif'])
disp.plot(cmap='Blues')
plt.title(
    'Confusion Matrix Intra-Annotator')
plt.show()
data_ambigu['LABEL'] = (
    data_ambigu['FINAL'])
data_valid_final = data_valid_final[
    ['NORMALISASI', 'LABEL']
]data_ambigu_final = data_ambigu[
    ['NORMALISASI', 'LABEL']]
#4. DATASET HYBRID dengan gabung dataset valid dan ambigu
dataset_final = pd.concat([
    data_valid_final,
    data_ambigu_final])
dataset_final = (
    dataset_final
    .reset_index(drop=True))
```

```
print(
    "Jumlah Dataset Final :",
    len(dataset_final))
final_count = dataset_final[
    'LABEL'
].value_counts()
print(final_count)
dataset_final.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/dataset_final.csv',
    index=False)
import matplotlib.pyplot as plt
def plot_distribusi_final(dataset_final):
    # urutan label
    urutan = [
        'positif',
        'netral',
        'negatif']
    # hitung jumlah tiap sentimen
    final_count = dataset_final[
        'LABEL'
    ].value_counts().reindex(
        urutan,
        fill_value=0)
    # warna tiap sentimen
    color_map = {
        'positif': 'green',
        'netral': 'gray',
```

```
        'negatif': 'red'}
    colors = [
        color_map[label]
        for label in final_count.index]
    # buat diagram batang
    ax = final_count.plot(
        kind='bar',
        figsize=(6,4),
        color=colors)
    # judul dan label
    plt.title(
        'Distribusi Dataset Hybrid')
    plt.xlabel('Sentimen')
    plt.ylabel('Jumlah Data')
    plt.xticks(rotation=0)
    # tampilkan jumlah di atas batang
    for i in ax.patches:
        ax.annotate(
            str(int(i.get_height())),
            (i.get_x() + i.get_width()/2,
                i.get_height()),
            ha='center',
            va='bottom')
    plt.show()
plot_distribusi_final(dataset_final)
#split data 80:20
from sklearn.model_selection import train_test_split
```

```
X = dataset_final['NORMALISASI']
y = dataset_final['LABEL']
X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.20,
    random_state=42,
    stratify=y)
print(
    "Jumlah Data Training :",
    len(X_train))
print(
    "Jumlah Data Testing :",
    len(X_test))
#presentase distribusi
# hitung persentase training
train_dist = (
    y_train
    .value_counts(normalize=True)
    * 100).round(2)
# hitung persentase testing
test_dist = (
    y_test
    .value_counts(normalize=True)
    * 100).round(2)
# jumlah data training
train_count = y_train.value_counts()
```

```
# jumlah data testing
test_count = y_test.value_counts()
# gabungkan tabel
distribusi_df = pd.DataFrame({
    'Jumlah Training': train_count,
    'Training (%)': train_dist,
    'Jumlah Testing': test_count,
    'Testing (%)': test_dist
})
# urutkan label
urutan = [
    'positif',
    'netral',
    'negatif']
# tambahkan total
total_row = pd.DataFrame({
    'Jumlah Training': [
        distribusi_df['Jumlah Training'].sum()
    ],
    'Training (%)': [
        distribusi_df['Training (%)'].sum()
    ],
    'Jumlah Testing': [
        distribusi_df['Jumlah Testing'].sum()
    ],
    'Testing (%)': [
        distribusi_df['Testing (%)'].sum()]
```

```
}, index=['TOTAL'])
# gabungkan total ke tabel
distribusi_df = pd.concat([
    distribusi_df,
    total_row])
# tampilkan tabel
distribusi_df
#gabungkan menjadi dataframe
train_df = pd.DataFrame({
    'NORMALISASI': X_train,
    'LABEL': y_train})
test_df = pd.DataFrame({
    'NORMALISASI': X_test,
    'LABEL': y_test})
train_df = train_df.reset_index(drop=True)
test_df = test_df.reset_index(drop=True)
train_df.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_training.csv',
    index=False)
test_df.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/data_testing.csv',
    index=False)

#ekstraksi fitur embedding IndoBERT
import torch
import numpy as np
from transformers import (
```

```
    AutoTokenizer,
    AutoModel)
model_name = 'indobenchmark/indobert-base-p1'
tokenizer = AutoTokenizer.from_pretrained(
    model_name)
model = AutoModel.from_pretrained(
    model_name)
#FUNGSI EMBEDDING INDO-BERT Menggunakan:CLS token embedding
def indoBERT_embedding(text):
# tokenisasi
    inputs = tokenizer(
        text,
        return_tensors='pt',
        truncation=True,
        padding=True,
        max_length=128)
    # nonaktifkan gradient
    with torch.no_grad():
        outputs = model(**inputs)
    # ambil CLS token
    embeddings = outputs.last_hidden_state[:,0,:]
    return embeddings.squeeze().numpy()
#embedding data training
X_train_embed = np.array([
    indoBERT_embedding(text)
    for text in X_train])
#embedding data testing
```

```
X_test_embed = np.array([
    indoBERT_embedding(text)
    for text in X_test])
#simpan embedding
np.save(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/X_train_embed.npy',
    X_train_embed)
np.save(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/X_test_embed.npy',
    X_test_embed)
#simpan label
y_train.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/y_train.csv',
    index=False)
y_test.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/y_test.csv',
    index=False)
#STANDARISASI scaling
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(
    X_train_embed)
X_test_scaled = scaler.transform(
    X_test_embed)
import pandas as pd
# Pilih dimensi yang akan ditampilkan
kolom = [0, 1, 2, 766, 767]
```

```
# Sebelum scaling
before = pd.DataFrame(
    X_train_embed[:5, kolom],
    columns=[
        'Dim_1',
        'Dim_2',
        'Dim_3',
        'Dim_767',
        'Dim_768'])
# Sesudah scaling
after = pd.DataFrame(
    X_train_scaled[:5, kolom],
    columns=[
        'Dim_1',
        'Dim_2',
        'Dim_3',
        'Dim_767',
        'Dim_768'])
# Gabungkan
comparison = pd.concat(
    [before, after],
    axis=1,
    keys=[
        'Sebelum Scaling',
        'Sesudah Scaling'])
comparison = comparison.round(4)
comparison
```

```
comparison.to_excel(
    "/content/drive/MyDrive/SKRIPSI/SKRIPSI2/Perbandingan_Embedding_Scaling.xlsx",
    index=True)
np.save(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/X_train_scaled.npy',
    X_train_scaled)
np.save(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/X_test_scaled.npy',
    X_test_scaled)
#hyperparameter tunning dengan gridsearchCV
from sklearn.svm import SVC
from sklearn.model_selection import (
    GridSearchCV)
svm_model = SVC()
# ruang parameter hyperparameter tuning
param_grid = {
    'C': [
        0.1,
        1,
        10,
    ],
    'kernel': [
        'linear',
        'rbf']}
grid_search = GridSearchCV(
    estimator=svm_model,
```

```
    param_grid=param_grid,
    cv=5,
    scoring='accuracy',
    verbose=1,
    n_jobs=-1)
#training gridsearch dengan data scaling
grid_search.fit(
    X_train_scaled,
    y_train)
#simpan hasil tunning
cv_result = pd.DataFrame(
    grid_search.cv_results_)
cv_result = cv_result[[
    'param_C',
    'param_kernel',
    'mean_test_score',
    'rank_test_score']]
cv_result
#training final dg best parameter
best_svm = grid_search.best_estimator_
best_svm.fit(
    X_train_scaled,
    y_train)
#SKENARIO KE 3 prediksi testing TANPA CW
y_pred = best_svm.predict(
    X_test_scaled)
hasil_prediksi = pd.DataFrame({
```

```
    'TEXT': X_test,
    'ACTUAL': y_test,
    'PREDICT': y_pred})
hasil_prediksi.head()
hasil_prediksi.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_prediksi_svm.csv',
    index=False)
#evaluasi prediksi klasifikasi model
from sklearn.metrics import (
    accuracy_score,
    classification_report,
    confusion_matrix)
#akurasi
accuracy = accuracy_score(
    y_test,
    y_pred)
print(
    "Accuracy :",
    accuracy)
# MCC
from sklearn.metrics import matthews_corrcoef
mcc3 = matthews_corrcoef(
    y_test,
    y_pred)
print("MCC :", round(mcc3,4))
#CLASSIFICATION REPORT
print(
```

```
    classification_report(
        y_test,
        y_pred))
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report
# Prediksi
y_pred = best_svm.predict(X_test_scaled)
# Hitung confusion matrix BARU
cm = confusion_matrix(
    y_test,
    y_pred,
    labels=['negatif', 'netral', 'positif'])
# Visualisasi
plt.figure(figsize=(6,5))
sns.heatmap(
    cm,
    annot=True,
    fmt='d',
    cmap='Blues',
    xticklabels=['negatif','netral','positif'],
    yticklabels=['negatif','netral','positif'])
plt.xlabel("Predicted Label")
plt.ylabel("Actual Label")
plt.title("Confusion Matrix SVM")
plt.show()
#CLASS WEIGHT
svm_model = SVC(
```

```
    class_weight='balanced')
#melihat bobot setiap sentimen
from sklearn.utils.class_weight import compute_class_weight
classes = np.unique(y_train)
weights = compute_class_weight(
    class_weight='balanced',
    classes=classes,
    y=y_train)
class_weights = dict(
    zip(classes, weights))
print(class_weights)
weight_df = pd.DataFrame({
    'Sentimen': classes,
    'Jumlah_Data': [
        y_train.value_counts()[c]
        for c in classes],
    'Bobot': weights})
weight_df

#SKENARIO KE 4 SVM dengan class weight
svm_final = SVC(
    C=10,
    kernel='rbf',
    class_weight='balanced')
#training
svm_final.fit(
    X_train_scaled,
```

```
    y_train)
#prediksi SVM
y_pred_balanced = svm_final.predict(
    X_test_scaled)
hasil_svm_balanced = pd.DataFrame({
    'TEXT': X_test,
    'ACTUAL': y_test,
    'PREDICT': y_pred_balanced})
hasil_svm_balanced.to_csv(
    '/content/drive/MyDrive/SKRIPSI/SKRIPSI2/hasil_svm_class_weight.csv',
    index=False)
#acuracy
from sklearn.metrics import accuracy_score
accuracy_balanced = accuracy_score(
    y_test,
    y_pred_balanced)
print(
    "Accuracy:",
    accuracy_balanced)
#MCC
mcc4 = matthews_corrcoef(
    y_test,
    y_pred_balanced)
print("MCC :", round(mcc4,4))
#classification report
from sklearn.metrics import classification_report
print(
```

```
    classification_report(
        y_test,
        y_pred_balanced))
#confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(
    y_test,
    y_pred_balanced)
print(cm)
plt.figure(figsize=(6,5))
sns.heatmap(
    cm,
    annot=True,
    fmt='d',
    cmap='Blues',
    xticklabels=[
        'negatif',
        'netral',
        'positif'
    ],
    yticklabels=[
        'negatif',
        'netral',
        'positif'])
plt.xlabel(
    'Predicted Label')
plt.ylabel(
```

```
    'Actual Label')
plt.title(
    'Confusion Matrix SVM + Class Weight')
plt.show()
#distribusi prediksi sentimen tanpa pembobotan kelas
pred_no_weight = pd.Series(
    y_pred
).value_counts()
distribusi_no_weight = pd.DataFrame({
    'Sentimen': pred_no_weight.index,
    'Jumlah': pred_no_weight.values})
distribusi_no_weight
#distribusi prediksi sentimen model dengan pembobotan kelas
pred_weight = pd.Series(
    y_pred_balanced
).value_counts()
pred_weight

# DATA NON-HYBRID dengan indoBERT
label_df
label_df.columns
#split data
from sklearn.model_selection import train_test_split
X = label_df['NORMALISASI']
y = label_df['LABEL']
X_train_auto, X_test_auto, y_train_auto, y_test_auto = train_test_split(
    X,
```

```
    y,
    test_size=0.2,
    random_state=42,
    stratify=y)
indoBERT_embedding(text)
print(
    "Jumlah Data Training :",
    len(X_train_auto))
print(
    "Jumlah Data Testing :",
    len(X_test_auto))
# jumlah data training
train_auto_count = y_train_auto.value_counts()
# jumlah data testing
test_auto_count = y_test_auto.value_counts()
# gabungkan tabel
distribusi_df = pd.DataFrame({
    'Jumlah Training': train_auto_count,
    'Training (%)': train_dist,
    'Jumlah Testing': test_auto_count,
    'Testing (%)': test_dist})
# urutkan label
urutan = [
    'positif',
    'netral',
    'negatif']
# tambahkan total
```

```
total_row = pd.DataFrame({
    'Jumlah Training': [
        distribusi_df['Jumlah Training'].sum()],
   'Training (%)': [
        distribusi_df['Training (%)'].sum()
    ],
    'Jumlah Testing': [
        distribusi_df['Jumlah Testing'].sum()
    ],
   'Testing (%)': [
       distribusi_df['Testing (%)'].sum()]
}, index=['TOTAL'])
# gabungkan total ke tabel
distribusi_df = pd.concat([
    distribusi_df,
    total_row])
# tampilkan tabel
distribusi_df
#embedding
X_train_embed_auto = X_train_auto.apply(indoBERT_embedding)
X_test_embed_auto = X_test_auto.apply(indoBERT_embedding)
import numpy as np
X_train_embed_auto = np.vstack(X_train_embed_auto)
X_test_embed_auto = np.vstack(X_test_embed_auto)
#scalling
from sklearn.preprocessing import StandardScaler
scaler_auto = StandardScaler()
```

```
X_train_scaled_auto = scaler_auto.fit_transform(
    X_train_embed_auto)
X_test_scaled_auto = scaler_auto.transform(
    X_test_embed_auto)
#class weight data non-hybrid
from sklearn.utils.class_weight import compute_class_weight
import numpy as np
classes = np.unique(y_train_auto)
weights = compute_class_weight(
    class_weight='balanced',
    classes=classes,
    y=y_train_auto)
class_weight_dict = dict(
    zip(classes, weights))
print(class_weight_dict)
weight_df = pd.DataFrame({
    'Sentimen': classes,
    'Jumlah_Data': [
        y_train_auto.value_counts()[c]
        for c in classes
    ],
    'Bobot': weights})
weight_df
#training SVM no class weight
from sklearn.svm import SVC
svm_auto = SVC(
    C=10,
```

```
    kernel='rbf')
svm_auto.fit(
    X_train_scaled_auto,
    y_train_auto)
# SKENARIO 1 NON HYBRID prediksi SVM TANPA CLASS WEIGHT
y_pred_auto = svm_auto.predict(
    X_test_scaled_auto)
from sklearn.metrics import accuracy_score
acc_auto = accuracy_score(
    y_test_auto,
    y_pred_auto)
print(
    "Accuracy Auto Labeling:",
    round(acc_auto,4))
#MCC
mcc1 = matthews_corrcoef(
    y_test_auto,
    y_pred_auto)
print("MCC :", round(mcc1,4))
from sklearn.metrics import classification_report
print(
#CLASIFICATION MATRIX
    classification_report(
        y_test_auto,
        y_pred_auto))
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

```
import matplotlib.pyplot as plt
# confusion matrix
cm = confusion_matrix(
    y_test_auto,
    y_pred_auto,
    labels=['negatif', 'netral', 'positif'])
# plot
plt.figure(figsize=(6,5))
sns.heatmap(
    cm,
    annot=True,
    fmt='d',
    cmap='Blues',
    xticklabels=['negatif', 'netral', 'positif'],
    yticklabels=['negatif', 'netral', 'positif'])
plt.title('Confusion Matrix SVM (Auto Labeling)')
plt.xlabel('Prediksi')
plt.ylabel('Aktual')
plt.show()
#SKENARIO KE 2 NON HYBRID SVM dengan class weight
from sklearn.svm import SVC
svm_cw = SVC(
    C=10,
    kernel='rbf',
    class_weight=class_weight_dict)
#training dengan scaling
svm_cw.fit(
```

```
    X_train_scaled_auto,
    y_train_auto)
#testing dengan SVM
y_pred_cw = svm_cw.predict(
    X_test_scaled_auto)
$AKURASI
from sklearn.metrics import accuracy_score
acc_cw = accuracy_score(
    y_test_auto,
    y_pred_cw)
print(
    f'Accuracy: {acc_cw:.4f}')
#MCC
mcc2 = matthews_corrcoef(
    y_test_auto,
    y_pred_cw)
print("MCC :", round(mcc2,4))
#CLASIFICATION REPORT
from sklearn.metrics import classification_report
print(
    classification_report(
        y_test_auto,
        y_pred_cw))
#CONFUSION MATRIX
cm_cw = confusion_matrix(
    y_test_auto,
    y_pred_cw,
```

```
    labels=[
        'negatif',
        'netral',
        'positif'])
plt.figure(figsize=(6,5))
sns.heatmap(
    cm_cw,
    annot=True,
    fmt='d',
    cmap='Blues',
    xticklabels=[
        'negatif',
        'netral',
        'positif'
    ],
    yticklabels=[
        'negatif',
        'netral',
        'positif'])
plt.title(
    'Confusion Matrix SVM + Class Weight')
plt.xlabel('Label Prediksi')
plt.ylabel('Label Aktual')
plt.tight_layout()
plt.show()
#evaluasi perbandingan 4 skenario
from sklearn.metrics import classification_report
```

```
import pandas as pd
#auto
report_auto = classification_report(
    y_test_auto,
    y_pred_auto,
    output_dict=True)
#auto+cw
report_auto_cw = classification_report(
    y_test_auto,
    y_pred_cw,
    output_dict=True)
#hybrid
report_hybrid = classification_report(
    y_test,
    y_pred,
    output_dict=True)
#hybrid+cw
report_hybrid_cw = classification_report(
    y_test,
    y_pred_balanced,
    output_dict=True)
hasil_perbandingan = pd.DataFrame({
    'Skenario': [
        'No-Hybrid + No Class Weight',
        'No-Hybrid + Class Weight',
        'Hybrid  + No Class Weight',
        'Hybrid + Class Weight'
```

```
],
'Accuracy': [
    acc_auto,
    acc_cw,
    accuracy,
    accuracy_balanced
],
'Precision': [
    report_auto['macro avg']['precision'],
    report_auto_cw['macro avg']['precision'],
    report_hybrid['macro avg']['precision'],
    report_hybrid_cw['macro avg']['precision']
],
'Recall': [
    report_auto['macro avg']['recall'],
    report_auto_cw['macro avg']['recall'],
    report_hybrid['macro avg']['recall'],
    report_hybrid_cw['macro avg']['recall']
],
'F1-Score': [
    report_auto['macro avg']['f1-score'],
    report_auto_cw['macro avg']['f1-score'],
    report_hybrid['macro avg']['f1-score'],
    report_hybrid_cw['macro avg']['f1-score']
],
'MCC': [
    mcc1,
```

```
        mcc2,
        mcc3,
        mcc4
    ]
}).round(4)
hasil_perbandingan


#WORD CLOUD
!pip install wordcloud
!pip install Sastrawi
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
from wordcloud import WordCloud
import matplotlib.pyplot as plt
from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory


# Stopword bawaan Sastrawi
stop_factory = StopWordRemoverFactory()
stopwords = set(stop_factory.get_stop_words())
# Custom stopword untuk data TikTok/MBG
custom_stopwords = {
    'tidak','di','ada','nya','jadi','buat','aja','nih','sih','dong',
    'kan','kok','lah','pun','iya','ya','yang',
    'ga','gak','udah','terus','semua',
    'orang','banget','kalo','kalau','juga','dari','pada','atau','utk','tapi','sekali',
    'harus','mau','si','kamu','lebih','bagi','cuma'}
```

```
# Gabungkan stopword bawaan + custom
stopwords.update(custom_stopwords)
# Fungsi stopword removal
def remove_stopword(text):
    words = str(text).split()
    words = [
        word
        for word in words
        if word.lower() not in stopwords]
    return ' '.join(words)
# Terapkan ke dataset
dataset_final['TEXT_STOPWORD'] = dataset_final[
    'NORMALISASI'
].apply(remove_stopword)
dataset_final[
    ['NORMALISASI','TEXT_STOPWORD']
].head(10)
#STEMMING
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
factory = StemmerFactory()
stemmer = factory.create_stemmer()
def stemming_text(text):
    return stemmer.stem(str(text))
dataset_final['TEXT_STEMMING'] = dataset_final[
    'TEXT_STOPWORD'
].apply(stemming_text)
dataset_final[
```

```
    ['TEXT_STOPWORD','TEXT_STEMMING']
].head(10)
import os
# folder penyimpanan word cloud
folder_save = "/content/drive/MyDrive/SKRIPSI/SKRIPSI2/WordCloud"
os.makedirs(
    folder_save,
    exist_ok=True)
sentimen_list = [
    'positif',
    'netral',
    'negatif']
plt.figure(figsize=(18,5))
for i, sentimen in enumerate(sentimen_list):
    text = ' '.join(
        dataset_final[
            dataset_final['LABEL']==sentimen
        ]['TEXT_STEMMING'].astype(str))
    wc = WordCloud(
        width=1000,
        height=500,
        background_color='white',
        collocations=False
    ).generate(text)
    plt.subplot(1,3,i+1)
    plt.imshow(
        wc,
```

```
        interpolation='bilinear')
    plt.axis('off')
    plt.title(
        f'Sentimen {sentimen.capitalize()}')
    # simpan masing-masing word cloud
    wc.to_file(
        f"{folder_save}/wordcloud_{sentimen}.png")
plt.tight_layout()
plt.show()
```

Lampiran 2

Kamus Inset *(Indonesia Sentiment)*

	NEGATIF	
No	word	weight
1	putus tali gantung	-2
2	gelebah	-2
3	gobar hati	-2
4	tersentuh (perasaan)	-1
5	isak	-5
6	larat hati	-3
7	nelangsa	-3
8	remuk redam	-5
9	tidak segan	-2
10	gemar	-1
11	tak segan	-1
12	sesal	-4
13	pengen	-2
14	penghayatan	-2
15	absorpsi	-1
16	linu	-4
17	salah benang	-1
18	sakit	-5
19	lara	-5
20	zuhud	-1
21	mencederai	-4
22	mengingkari	-4
23	maaf	-3
24	mengkhianat	-4
25	mencelakai	-5
26	mulu	-1
27	ngga	-2
28	borong	-1
29	lever	-2
30	kasian	-3
31	gamau	-4
32	doang	-1
33	pulas	-1
34	abis	-2
35	coba	-1
36	kangen	-3
37	kalau	-1
38	maunya	-1
39	seandainya	-1
40	marilah	-1
41	bener	-1
42	yaudah	-4
43	nggak	-3
44	gatau	-1
45	apaan	-4
46	ngakak	-2
47	atuh	-1
48	sekali	-1
49	menarik hati	-1
50	cedayam	-2
51	kece	-3
52	termakan	-1
53	belom	-1
54	malem	-1
55	mencekau	-2
56	menduga	-1
57	menyuarakan	-1
58	memprediksi	-1
59	membunyikan	-1
60	menerka	-1
61	menaksir	-1
62	mengantisipasi	-1
63	nangis	-5
64	rompok	-2
65	soalnya	-1
66	griya	-1
67	gubuk	-2
68	anjir	-3
69	putus jiwa	-5
70	berlalu dr dunia	-5

	POSITIF	
No	word	weight
1	hai	3
2	merekam	2
3	ekstensif	3
4	paripurna	1
5	detail	2
6	pernik	3
7	belas	2
8	welas	4
9	kabung	1
10	rahayu	4
11	maaf	2
12	hello	2
13	promo	3
14	terimakasih	5
15	cover	3
16	mohon	2
17	mengawal	2
18	statistik	1
19	keluangan	3
20	jalan terbuka	3
21	banyaknya	3
22	lebar	3
23	bentang	1
24	hendaknya	1
25	silahkan	3
26	semboyan	2
27	ditunggu	2
28	akses	2
29	penerangan	2
30	hi	1
31	dibantu	2
32	makasih	4
33	halo	1
34	thanks	3
35	pengembangan	3
36	diva	2
37	punya	3
38	tidak segan	2
39	detailnya	1
40	tak segan	2
41	aktivasi	2
42	asih	3
43	kasih sayang	5
44	kekaguman	4
45	kehangatan	4
46	afeksi	2
47	renjana	2
48	amor	2
49	cinta kasih	5
50	tresna	2
51	filantropi	2
52	cintrong	2
53	suasana (hati)	1
54	dinamika	3
55	tuhan	3
56	merespon	3
57	makmur	4
58	suka cita	4
59	pengguna	1
60	tunggu	1
61	lotre	2
62	nggak	1
63	kupon	3
64	terpelihara	4
65	terawat	5
66	tersadar	3
67	tari	1
68	gejolak	1
69	kejutan	3
70	pesta	4

Lampiran 3

***Link* Dataset, Kamus InSet dan Hasil Analisis**

https://drive.google.com/drive/folders/1PIBMhA3hgtkXOuh2XC4UlvulP-C0f7sh?usp=sharing

Lampiran 4

Kartu Bimbingan

UNIVERSITAS PUTRA BANGSA

Kampus Pusat : Jl. Ronggowarsito 18 Pejagoan Kebumen, Telp. 0287-384011
Kampus Dua : Jl. Raya Buntu - Gombong KM 05 Kemranjen Banyumas, Telp. 0287-5296662

Kartu Bimbingan Skripsi

Nama	:	Ami Nafisatun Najah
NIM	:	220202619
Program Studi	:	Sarjana Ilmu Komputer
Fakultas	:	Sains dan Teknologi
Dosen Pembimbing	:	Wahyuni Windasari, S.Si., M.Sc.

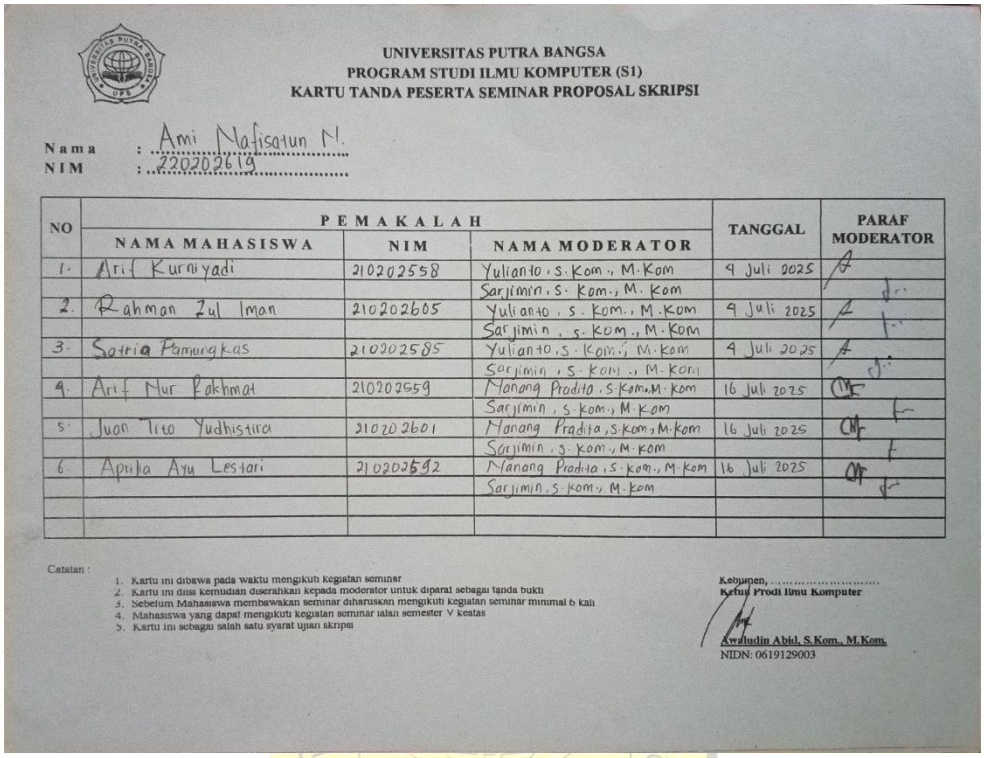
No	Tanggal	Materi Bimbingan	Status
1	08 November 2025	Konsultasi terkait tema, alur, dan judul skripsi	Valid
2	22 November 2025	Pencarian Dataset dan Pengarahan ke BAB 1	Valid
3	20 Desember 2025	Konsultasi BAB 1 terkait isi dan sistematika penulisan, serta lanjut untuk BAB 2	Valid
4	29 Januari 2026	Bimbingan BAB 2 terkait landasan teori yang dipakai dan penelitian terdahulu atau gap penelitian	Valid
5	25 Februari 2026	BAB 3 terkait isi dan tahapan penelitian	Valid
6	27 Februari 2026	Kelengkapan proposal penelitian dari BAB 1,2,3 dan daftar pustaka	ACC Seminar
7	21 Mei 2026	Diskusi Alur Pengolahan Data dan kelengkapan isi BAB 4	Valid
8	17 Juni 2026	Diskusi terkait hasil pengolahan data dan penulisan BAB 4	Valid
9	30 Juni 2026	Diskusi terkait hasil pengolahan data pada BAB 4 dan penyususnan narasi hasil	Valid
10	13 Juli 2026	Pembahasan dan penulisan BAB 4 hasil penelitian dan BAB 5 kesimpulan dan saran	Valid
11	23 Juli 2026	Presentasi PPT Penelitian, masukan dan arahan terkait presentasi	Valid
12	27 Juli 2026	Kelengkapan isi skripsi dari BAB 1,2,3,4,5, daftar pustaka dan lampiran	ACC Sidang

Kebumen, 28 Juli 2026
Dosen Pembimbing

Wahyuni Windasari, S.Si., M.Sc.
NIDN. 0611068804

Lampiran 5

Kartu Sempro



UNIVERSITAS PUTRA BANGSA
PROGRAM STUDI ILMU KOMPUTER (S1)
KARTU TANDA PESERTA SEMINAR PROPOSAL SKRIPSI

Nama : Ami Nafisatun N.
NIM : 22020261 9

NO	PEMAKALAH			TANGGAL	PARAF MODERATOR
	NAMA MAHASISWA	NIM	NAMA MODERATOR		
1.	Arif Kurniyadi	210202558	Yulianto, S.Kom., M.Kom	9 Juli 2025	
			Sarjimin, S. Kom., M. Kom		
2.	Rahman Zul Iman	210202605	Yulianto, S. Kom., M.Kom	9 Juli 2025	
			Sarjimin, S. Kom., M.Kom		
3.	Satria Pamungkas	210202585	Yulianto, S. Kom., M. Kom	9 Juli 2025	
			Sarjimin, S. Kom., M. Kom		
4.	Arif Nur Rakhmat	210202559	Manang Pradita, S.Kom, M.Kom	16 Juli 2025	
			Sarjimin, S. Kom., M.Kom		
5.	Juan Tito Yudhistira	210202601	Manang Pradita, S.Kom, M.Kom	16 Juli 2025	
			Sarjimin, S. Kom., M.Kom		
6.	Aprila Ayu Lestari	210202592	Manang Pradita, S. Kom., M.Kom	16 Juli 2025	
			Sarjimin, S. Kom., M.Kom		

Catatan :
1. Kartu ini dibawa pada waktu mengikuti kegiatan seminar
2. Kartu ini diisi kemudian diserahkan kepada moderator untuk diparaf sebagai tanda bukti
3. Sebelum Mahasiswa membawakan seminar diharuskan mengikuti kegiatan seminar minimal 6 kali
4. Mahasiswa yang dapat mengikuti kegiatan seminar ialah semester V keatas
5. Kartu ini sebagai salah satu syarat ujian skripsi

Kebumen, ...
Ketua Prodi Ilmu Komputer

Awaludin Abid, S.Kom., M.Kom.
NIDN: 0619129003