**Summary of base paper:**

By using Conditional Generative Adversarial Networks(CGAN), Razaque et al.’s study”Twitter Sentiment Analysis using Conditional GAN” presents a novel method for sentiment categorization on Twitter data. In order to get enough accuracy conventional technique like support vector machines and deep learning models like convolutional neural network and long short term memory networks fee quintly need a significant amount of labelled data. By teaching a discriminator to discern between created and actual samples and predict sentiment labels this CGAN architecture overcomes the issue of data scarcity. A generator is a trained to generate synthetic feature vectors conditioned on sentiment labels within the CGAN framework, the approach integrates and lcm discriminator with CNN based feature extraction. According to experimental results pertaining To the US election of 2020, the cgg in performs better than traditional classifier, pithan accuracy of almost 93%, which is a notable increases above the CNN and LSTM baseline models. The paper empathiser how generative adversarial learning may increases resilience , decrease resilience on massive manually labelled data sets, and improve text classification performance. The study significant because it offers a viable avenue for future research in natural language processing problems and shows how adversarial training may efficiently supplement sparse data.

**Introduction:**

The conditional GAN method for Twitter sentiment analysis put out by Razaque et al.2024 is duplicated and simplified in this project. The goal is to use flux deep learning package to create a repeatable implementation of cspn based classifier in Julia. This study is important because in situations when labelled data is inadequate, typical supervisor learning techniques frequently so far from over fitting and restricted generalization. To study intends to increase classifier robustness and provide the groundwork for future advancements in adversarial learning methods in next classification by integrating synthetic feature creation condition on sentiment labelling.

**Methodology:**

Data preparation, baseline classifier training and conditional GAN generation Who are the 3 primary steps in our methodology. Average step was intended to show a whole pipeline from raw data to generate to modelling and to build upon the one before it.

**Preprocessing:**

We started by writing preprocessing.jl, which encodes each tweet into a series of numbers that represent word indices, tokenizes tweets into individual words, and creates a vocabulary of distinct tokens. A binary format was used to map the labels: 1 denoted positive emotion and 0 denoted negative sentiment. All treated data was stored in preprocessed\_data.jld2 to provide uniformity between trials. This method eliminates the requirement to perform the preprocessing each time the same dataset is imported.

**Classifier:**

In train\_classifier.jl, we then constructed a supervised classifier. The model design comprised two dense layers with a sigmoid activation to predict the likelihood of positive sentiment, an embedding layer to learn word representations, and a mean pooling layer to summarize sequence information. We used the Adam optimizer and binary cross-entropy loss to train this model across ten epochs. This baseline was used to understand the classifier's performance in the absence of any artificial data augmentation.

Epoch 1/10 | Loss = 0.7096

Epoch 2/10 | Loss = 0.7076

Epoch 3/10 | Loss = 0.7056

Epoch 4/10 | Loss = 0.7036

Epoch 5/10 | Loss = 0.7016

Epoch 6/10 | Loss = 0.6997

Epoch 7/10 | Loss = 0.6978

Epoch 8/10 | Loss = 0.6959

Epoch 9/10 | Loss = 0.6941

Epoch 10/10 | Loss = 0.6923

**Conditional GAN:**

In gan.jl, we created a Conditional GAN (CGAN) for the generating portion. The generator learns to create synthetic embeddings that mimic actual tweet representations by using label information and random noise as input. The discriminator, which is likewise conditioned on the target label, attempts to differentiate between actual embeddings and those produced by the model. In order to verify that the discriminator and generator could learn to deceive one another in a stable manner, we first trained this CGAN using synthetic data. The model was prepared for integration with actual embeddings obtained from the preprocessed dataset once the training dynamics were confirmed.

Epoch 1/50 | D\_loss = ~1.2 | G\_loss = ~0.9

...

Epoch 50/50 | D\_loss = ~0.6 | G\_loss = ~0.7

At last I have done cross validation too to improve robustness and estimate variance trained for 10 epoch per fold with adam optimizer, embedding dimensions 8, hidden layer 16, and full batch , our method concentrated on thick layers, as opposed to the original paper's usage of CNN and LSTM layers to capture deeper temporal patterns and local relationships in text. This decision was taken specifically to guarantee that the code could execute consistently on many computers, simplify the design, and lower the computing needs. Although the utmost precision that may be achieved may be limited by this simplification, it made it possible for us to clearly illustrate the conditional generation idea and procedure.

**Results :**

Ten-fold cross-validation was used to assess the baseline classifier's generalization performance. Three samples for validation and 27 samples for training were utilized in each fold. This method helps guarantee that the model is evaluated throughout the whole dataset, offering a reliable performance indicator.

Starting 10-fold cross-validation...

🟢 Fold 1: Training 27 samples, Validation 3 samples

✅ Fold 1 Validation Accuracy: 66.67%

🟢 Fold 2: Training 27 samples, Validation 3 samples

✅ Fold 2 Validation Accuracy: 66.67%

🟢 Fold 3: Training 27 samples, Validation 3 samples

✅ Fold 3 Validation Accuracy: 66.67%

🟢 Fold 4: Training 27 samples, Validation 3 samples

✅ Fold 4 Validation Accuracy: 100.0%

🟢 Fold 5: Training 27 samples, Validation 3 samples

✅ Fold 5 Validation Accuracy: 66.67%

🟢 Fold 6: Training 27 samples, Validation 3 samples

✅ Fold 6 Validation Accuracy: 100.0%

🟢 Fold 7: Training 27 samples, Validation 3 samples

✅ Fold 7 Validation Accuracy: 0.0%

🟢 Fold 8: Training 27 samples, Validation 3 samples

✅ Fold 8 Validation Accuracy: 33.33%

🟢 Fold 9: Training 27 samples, Validation 3 samples

✅ Fold 9 Validation Accuracy: 66.67%

🟢 Fold 10: Training 27 samples, Validation 3 samples

✅ Fold 10 Validation Accuracy: 33.33%

🏁 10-Fold Cross-Validation Average Accuracy: 60.0%

All folds had an average validation accuracy of 60%, which indicates a reasonable level of predictive potential. Although accuracy varies greatly between folds because to the short amount of instances, these findings demonstrate that a basic embedding-based classifier may do rather well on this tiny dataset even in the absence of CNNs or LSTMs.  
  
Training on synthetic data was successful for the Conditional GAN, and convergence was seen in the discriminator and generator losses. Although this project phase did not include a quantitative evaluation of the GAN outputs on actual embeddings, the implementation showed that the CGAN framework can provide class-conditional samples and may be expanded for augmentation in further work.

Conclusion :

This research showed how to replicate the entire Twitter sentiment classification pipeline workflow in Julia, including conditional GAN moulding, baseline supervisor learning and preprocessing. In order to prioritize repeatability and clarity over maximum speed our version of the article proposed fully adapted a simpler architecture.

The classifier demonstrated that even lightweight dense models can learn significant pattern from tokenised text by achieving an average accuracy of 60% during 10 fold cross validation. In order to build the groundwork for future advancement like training on actual embedding and incorporating produced samples into classifier training for augmentation, the conditional GAN implementation confirmed that the generator and discriminator can be taught to represent class conditional distributions.

Overall the study empathises the need of through preprocessing careful architecture section and rigorous assessment while highlighting the possibilities and difficulties of text categorization and generate modelling low resource situations,

GITHUB LINK:

https://github.com/ramanmannem17/conditional-gan-replication.git

References:

Mahalakshmi, V., Shenbagavalli, P., Raguvaran, S., Rajakumareswaran, V. and Sivaraman, E., 2024. Twitter sentiment analysis using conditional generative adversarial network. *International Journal of Cognitive Computing in Engineering*, *5*, pp.161-169.