Automatic annotation of open-ended responses

Ziyue Liu (PhD, C11)

July 10, 2025 Duisburg Retreat

How much does it cost you (e.g., in time and money) to annotate your textual data (open-ended responses)?

01 Background

Why do we use auto-annotation?

- Reduce cost: the per-annotation cost of ChatGPT is less than $0.003 (Gilardi et al., 2023)
- Fast speed

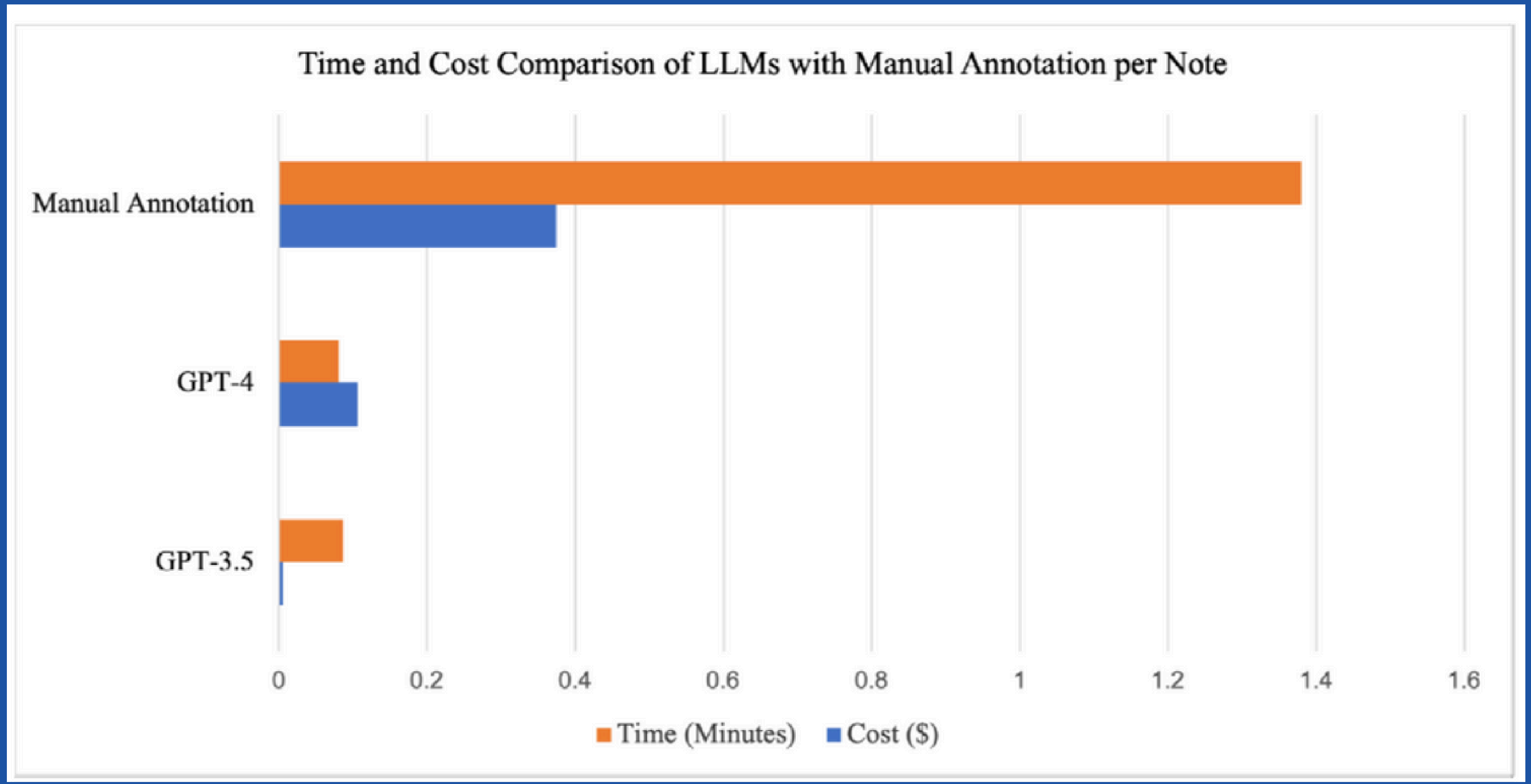


Fig 1: Cost and time comparison of GPT-4 and GPT-3.5 with manual annotation estimate per note (Ralevski et al, 2024)

01 Background

Why do we use auto-annotation?

- Reduce cost: the per-annotation cost of ChatGPT is less than $0.003 (Gilardi et al., 2023)
- Fast speed

What factors can affect auto-annotation?

- The quality of manually coded data (e.g., ambiguous texts, human errors)
- Trained models (e.g., data sizes, data types)

02 Methods

How do we improve the performance of auto-annotation?

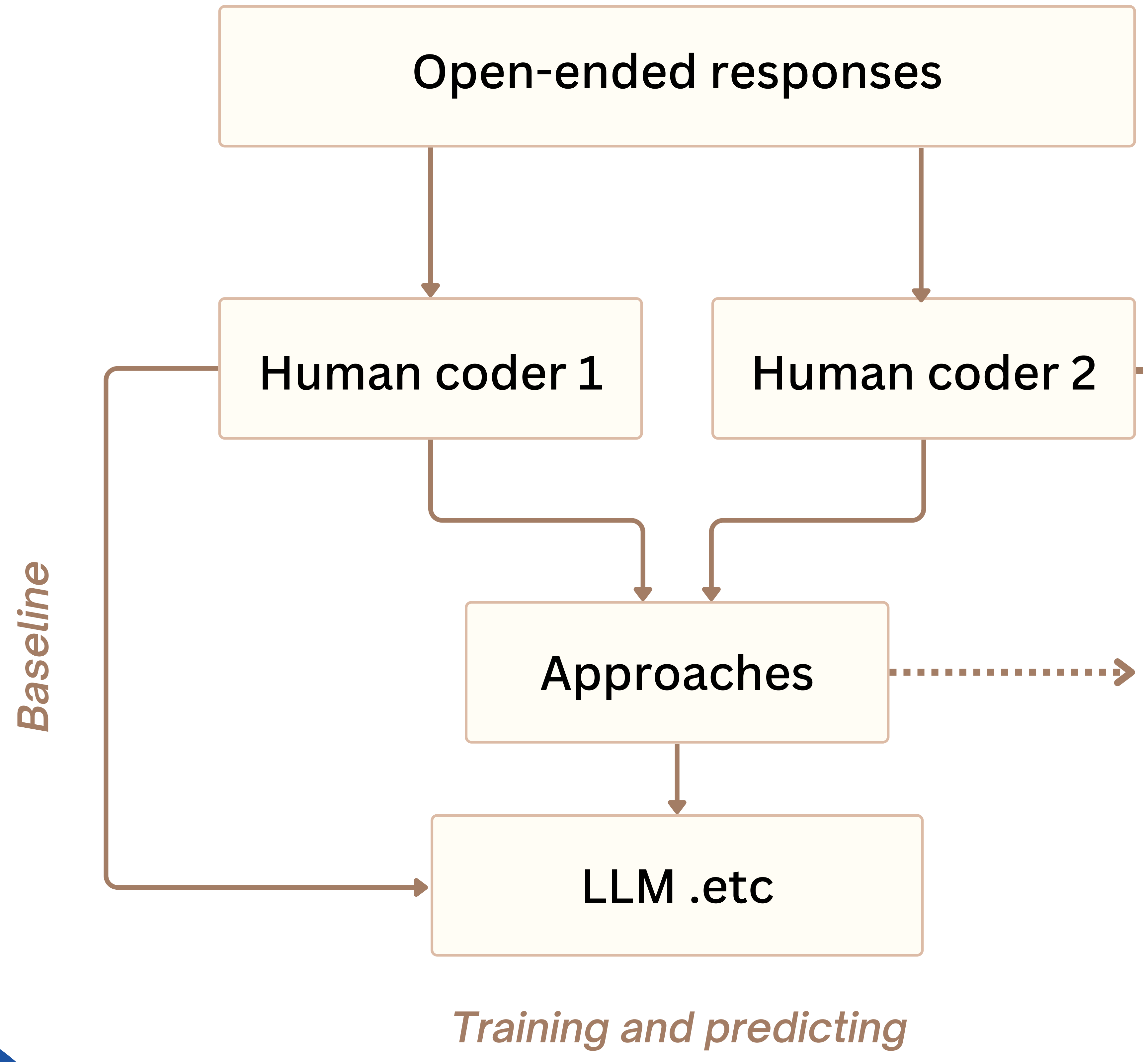
Improve manually coded data quality: double-coded data

Carefully select models

How do we treat double-coded data in auto-annotation?

“Replicate”, “Remove differences”, “Expert resolves”

(Schonlau & He, 2020)



"Replicate": duplicate each response regardless of whether the two annotations are identical or different.

"Remove differences": remove responses with disagreements

"Expert resolves" : invite an expert to re-annotate the responses with disagreements

(Schonlau & He, 2020)

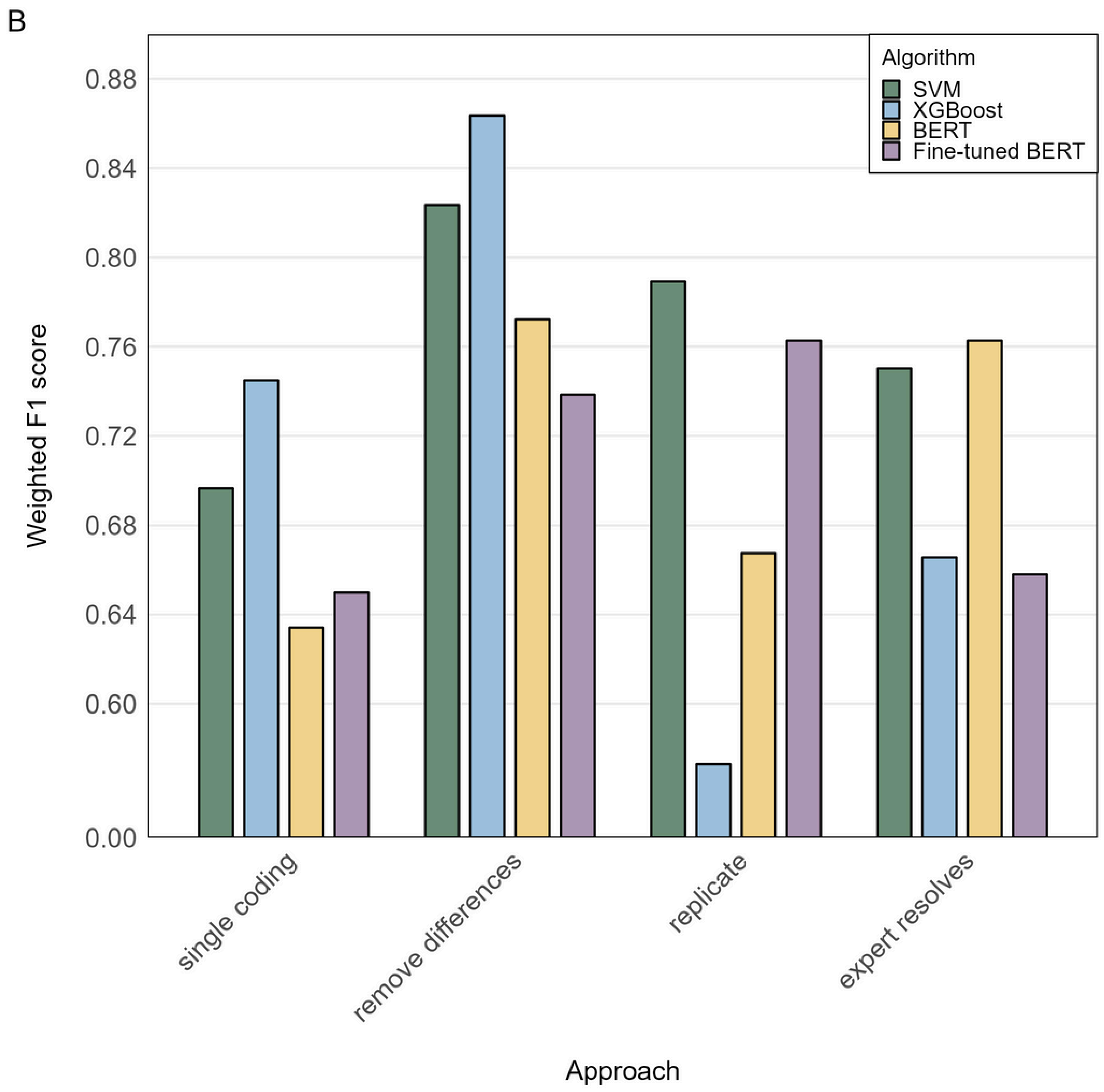
03 Workflow

Apply approach (only necessary in my case)

→ Split data (train & test) → NLP → Train model → Predict & Evaluate

Split data (train & test)	NLP	Train model	Predict & Evaluate
Consider the unbalanced issue, 80% + 20%	Tokenize, Create Corpus, Create document term matrix, R packages: quanteda	Import model, Set parameters, Train on the training dataset Overfit issue	Predict on the test dataset, Accuracy, F1-score

04 Results



Training time: ca. 1 minute with SVM and XGBoost, ca.3 hours with BERT (CPU) for ca. 1500 responses.

If the data is double-coded, applying the remove differences approach and XGBoost are recommended, and SVM can also be considered.

If the data is single-coded, XGBoost is suggested.

BERT (a large language model) is not recommended in this case.

References

Gilardi, F., Alizadeh, M., & Kubli, M. (2023). ChatGPT outperforms crowd workers for text-annotation tasks. Proceedings of the National Academy of Sciences, 120(30), e2305016120.

Ralevski, A., Taiyab, N., Nossal, M., Mico, L., Piekos, S. N., & Hadlock, J. (2024). Using Large Language Models to Annotate Complex Cases of Social Determinants of Health in Longitudinal Clinical Records. medRxiv.

He, Z., & Schonlau, M. (2020). Automatic coding of text answers to open-ended questions: Should you double code the training data? Social Science Computer Review, 38(6), 754-765.

He, Z., & Schonlau, M. (2020, August). Automatic coding of open-ended questions into multiple classes: whether and how to use double coded data. In Survey Research Methods (Vol. 14, No. 3, pp. 267-287).

Appendix A

An example code for automatic annotation using XGBoost model

```
# create corpus
corp_train <- corpus(df_train, text_field = "text")
corp_test <- corpus(df_test, text_field = "text")
# document term matrix
Dfm_train <- corp_train %>%
  tokens(remove_punct = TRUE, remove_numbers = TRUE, remove_symbols = TRUE, remove_separators = TRUE) %>%
  tokens_remove(stopwords::stopwords("da", source = "snowball")) %>%
  tokens_wordstem() %>%
  tokens_ngrams(n = 1) %>%
  dfm()
Dfm_test <- corp_test %>%
  tokens(remove_punct = TRUE, remove_numbers = TRUE, remove_symbols = TRUE, remove_separators = TRUE) %>%
  tokens_remove(stopwords::stopwords("da", source = "snowball")) %>%
  tokens_wordstem() %>%
  tokens_ngrams(n = 1) %>%
  dfm()
# using matched dfm
Dfm_matched <- dfm_match(Dfm_test, features=featnames(Dfm_train))
# xgb.DMatrix
ctrain <- xgb.DMatrix(Matrix(data.matrix(Dfm_train[,!colnames(Dfm_train) %in% c('label')])), label = as.numeric(Dfm_train$label)-1)
ctest <- xgb.DMatrix(Matrix(data.matrix(Dfm_matched[,!colnames(Dfm_matched) %in% c('label')])), label = as.numeric(Dfm_matched$label)-1)
colnames(ctest) <- NULL
```

Appendix A

An example code for automatic annotation using XGBoost model

```
# train the model
watchlist <- list(train = ctrain, test = ctest)
xgb_params <- list("objective" = "multi:softmax",
                   "eval_metric" = "mlogloss",
                   "num_class" = 4,
                    "nrounds" = 50)
xgbmodel <- xgboost(params = xgb_params,
                    data = ctrain,
                    nfold = 30,
                    nrounds = 50)
# prediction and evaluation
xgbmodel.predict <- predict(xgbmodel, newdata = ctest)
#confusion matrix
ts_label <- as.numeric(df_test$label)-1
ts_label <- as.factor(ts_label)
xgbmodel.predict <- as.factor(xgbmodel.predict)
cm <- confusionMatrix(xgbmodel.predict, ts_label)
```

Appendix B

Approach for dealing with double-coded data

Replicate

Duplicate each text response in the training data, including each coding instance, regardless of whether the two codes are identical or different.

Text	Coder 1	Coder 2
Text 1	positive	negative
Text 2	positive	positive

Text	Coder 1	Coder 2
Text 1	positive	negative
Text 1	positive	negative
Text 2	positive	positive
Text 2	positive	positive

Text	Label
Text 1	positive
Text 1	negative
Text 2	positive
Text 2	positive

Appendix B

Approach for dealing with double-coded data

Remove Differences

Remove text responses from the training data if the two coders coded them differently.

Text	Coder 1	Coder 2
Text 1	positive	negative
Text 2	positive	positive

Text	Coder 1	Coder 2
Text 2	positive	positive

Text	Label
Text 2	positive

Appendix B

Approach for dealing with double-coded data

Expert Resolves

Invite an expert to code the texts that disagree with the two coders.

Text	Coder 1	Coder 2
Text 1	positive	negative
Text 2	positive	positive

Text	Coder 1	Coder 2	Expert
Text 1	positive	negative	neutral
Text 2	positive	positive	

Text	Label
Text 1	neutral
Text 2	positive

Thank you!