DIKU NLP Course 2025: Group Project

Mahdi Mohammadzadeh Hung Quoc Pham Jonasz Witczak

1 Week 36

Week 36 focuses on understanding the dataset and establishing a transparent heuristic baseline. We (i) report dataset statistics overall and by language, (ii) extract the most frequent words in training *questions* for Arabic, Korean, and Telugu, and (iii) implement a rule-based classifier that decides if a question is answerable using only the question and its English context.

1.1 Dataset and Preprocessing

We use the combined course dataset based on TyDi QA / XOR RC / XOR-AttriQA with train/validation splits.

Tokenization for counts. We tokenize with a regex over non-whitespace spans (`\S+`) and drop punctuation-only tokens. The decision to use regex tokenization instead of language-specific tokenizers is motivated by comparability: it avoids introducing language-specific heuristics that may bias counts toward some scripts more than others. While this choice reduces linguistic precision (e.g., failing to split compounds in Korean or to capture inflectional morphemes in Arabic), it ensures that reported statistics are consistent across languages and can be reproduced with minimal tooling.

MT for the rule-based baseline. Questions are translated to English using NLLB-200 ("facebook/nllb-200-distilled-600M") before scoring; if MT is not available on a given run, the original question text is used. NLLB-200 was selected because it supports all three target languages with a single model, allowing us to apply the same pipeline regardless of source language. Translation reduces the impact of script differences and enables our overlap-based heuristic to operate in a unified space. This design sacrifices some fidelity, as machine translation may normalize or paraphrase expressions, but it improves the feasibility of implementing a single, transparent baseline.

1.2 Data Statistics

We first report overall sizes and then per-language breakdowns.

1.3 Overall

The dataset contains the following numbers:

Split	#Examples	Word count (Q+C)
Train	15,343	1,603,888
Validation	3,011	317,922

Table 1: Overall dataset sizes and total word counts (questions + contexts).

1.3.1 Per language

The language-specific sizes and word counts are:

Lang	#Train	Words (Train)	#Valid	Words (Valid)
ar	2,558	283,247	415	45,881
ko	2,422	247,179	356	35,873
te	1,355	127,169	384	43,372

Table 2: Per-language sizes and word counts (question+context).

The distributions are imbalanced: Arabic and Korean contribute substantially more training examples than Telugu. However, Telugu validation contains relatively more examples, which could impact how performance metrics compare across languages. Word counts scale proportionally with number of examples, showing that the dataset is internally consistent.

1.4 Most Frequent Question Words

Top-5 in *training* questions per language (regex tokenization; punctuation-only tokens removed). Script text is wrapped in dedicated font commands to render correctly with XeLaTeX.

Lang	Word	Count	English gloss
ar	في	592	in
ar	من	584	from / of
ar	متى	535	when
ar	ما	441	what
ar	هو	349	is / he
ko	가장	527	most
ko	무엇인가?	495	what?
ko	언제	336	when
ko	몇	234	how many
ko	어디인가?	228	where?
te	ఎవరు?	186	who?
te	ఎన్ని	161	how many
te	ఎప్పుడు	151	when
te	ఏ	142	which / what
te	ఏది?	117	which one? / what?

Table 3: Top-5 most frequent words in training *questions* per language.

The frequent-word lists are dominated by interrogatives ("what," "when," "who") and light function words (e.g., Arabic في, Korean 가장). This pattern reflects the natural structure of QA datasets, where questions are usually short, formulaic, and rely heavily on interrogative forms. The presence of counting-related terms (e.g., Korean 몇 "how many," Telugu ఎన్ని "how many") indicates that numerical reasoning is a common task type, which has direct implications for model design. Rule-based baselines that rely solely on lexical overlap are likely to struggle with such questions, as numbers often require normalization or reasoning across multiple spans.

1.5 Rule-based Answerability

1.5.1 Design

We classify a question as *answerable* if there is sufficient surface evidence in the English context: (1) token overlap between question and context (after optional MT to English), with a slightly higher threshold for longer questions; (2) a digit overlap shortcut lowers the threshold when numbers/dates clearly match.

This design prioritizes transparency and ease of reproduction. The thresholding mechanism is intended to balance short factual questions against longer, more descriptive ones. The digit overlap rule reflects a linguistic observation: many QA pairs in this dataset involve years, counts, or dates, where matching numerals are highly indicative of answerability.

1.5.2 Implementation (summary)

Lowercased regex tokenization (\S+), remove punctuation-only tokens, deduplicate question tokens, compute overlap ratio against the set of context tokens, and apply a threshold with a digit-overlap adjustment. MT uses NLLB-200 with language codes ar→arb_Arab, ko→kor_Hang, te→tel_Telu, target eng_Latn (when the model is available).

1.5.3 Validation results

Lang	Accuracy	Precision	Recall	F1
ar	0.1422	1.0000	0.0193	0.0378
ko	0.0758	1.0000	0.0237	0.0464
te	0.2630	0.6818	0.0515	0.0958

Table 4: Rule-based answerability on the validation set.

1.6 Analysis of Results

The rule-based classifier achieves very high precision but extremely low recall. For Arabic and Korean, the model predicts almost exclusively "unanswerable," only flagging items as answerable when surface overlap is overwhelming. This explains the precision of 1.0 but recall below 0.03: the classifier never makes a false positive, but misses almost all true positives. Telugu shows slightly better balance (recall 0.05, F1 ≈ 0.10). This may be linked to the higher proportion of explicit lexical matches in Telugu contexts, or to translation artifacts that preserve surface forms more consistently than in Arabic or Korean. The poor performance overall demonstrates that pure token overlap is insufficient, especially across morphologically rich or script-diverse languages.

1.7 Discussion and Limitations

The overlap-based baseline is deliberately conservative. It establishes a transparent reference point, but the cost of transparency is coverage: high precision comes at the expense of recall. The method cannot handle morphological variants (e.g., Arabic case endings), synonyms, or paraphrase, and it is sensitive to machine translation errors. Digits partially mitigate this problem but only in narrow cases such as dates and numbers. These limitations suggest that future improvements should incorporate term-weighting schemes (BM25, TF–IDF) or supervised classifiers that capture semantic similarity beyond exact matches. Nonetheless, the current baseline provides a reproducible, language-

agnostic reference against which such models can be evaluated.

2 Week 37

2.1 Unigram Language Model

An implementation of a unigram language model was made with Laplace smoothing, following the n-gram models. Tokenization used the same approach from Week 36 for comparability,, and the model training was split for all questions in Arabic, Korean, and Telugu, along with the English contexts. The performance was evaluated on the validation split using perplexity. To be concrete, for each text in the validation set we first compute the log probability of its tokens under the unigram distribution. Then, as the log probabilities were accumulated, it was then normalized based on the number of tokens.

Language / Text Type	Validation Perplexity
Arabic questions	1851.10
Korean questions	1590.77
Telugu questions	1178.76
English contexts	3146.85

The unigram model overall gives off high perplexities, which is in essence expected as unigrams ignore word order and dependencies which are longer. Looking at the languages, we can see that Telugu achieves the lowest perplexity while Arabic and Korean score considerably higher. A possible reason for this is that the Telugu questions at least in this dataset are on average (from first glance) shorter and use a smaller range of vocabulary, which is perfect for a unigram distribution to capture.

2.2 Bigram Language Model

A bigram model was trained over tokenized text with explicit sentence boundaries (<s>, </s>) using add-1 (Laplace) smoothing. This setup captures short-range dependencies ignored by the unigram model.

Language	Bigram VP	Trigram VP
Arabic (ar) questions	1550.00	1767.65
English (en) contexts	17955.79	71213.59
Korean (ko) questions	1224.85	2575.33
Telugu (te) questions	920.02	1526.71

Table 5: Validation perplexities (VP) for Bigram/Trigram language models.

Bigram modeling notably reduces perplexity for English and Telugu but yields smaller or inconsistent gains elsewhere, likely due to limited data and the effects of smoothing on rare contexts. It effectively models short-range dependencies yet still struggles with sparse patterns.

2.3 Trigram Language Model

The trigram model, using two start symbols and the same Laplace smoothing, underperforms bigrams and even unigrams in several cases due to data sparsity and over-smoothing. More advanced smoothing or backoff methods, such as interpolated Kneser–Ney or Witten–Bell, would be required for higher-order n-grams to generalize effectively.

2.4 Trigram Language Model

We extend to trigrams with two start symbols. With vocabulary size V, we use add-1 smoothing.

2.4.1 Results and Discussion

As shown in Table 5 and **??**, our plain Laplace-smoothed trigram model underperforms bigrams and even unigrams in several cases (e.g., English contexts and Korean questions), indicating data sparsity and over-smoothing at order 3. Standard remedies would include interpolated Kneser–Ney or Witten–Bell smoothing and explicit back-off, which typically yield large gains for higher-order n-grams on limited data. Higher-order context helps only with appropriate smoothing/backoff; with naive add-1, trigrams are too sparse to generalize well on our training data.

3 Week 38

3.1 Linear SVM classifier

One of the models for classification that we chose, was the linear SVM model. For this we used TF-IDF, as it is a strong and simple baseline for text classification. It captures word importance, whilst doing a good job of excluding or down-weighing very common words in languages. Here we allowed for use of uni-, bi-, and trigrams, as they allow for the classifier to look at short phrases, which is useful for certain linguistic cues. For example, 'which year' etc.

We chose linear SVM as one of our classifiers, as SVM optimizes for maximum margins rather than probability estimations, which is different to the logistic classifier we also implemented. This can

give better generalization in sparse feature spaces like TF-IDF, which we are using. SVMs are also less prone to overfitting on noisy text data compared to tree-based models.

3.1.1 Evaluation

Table 6: Linear SVM (TF–IDF) results on the validation set. Per-class precision (P), recall (R), and F1 are reported for *Impossible* and *Answerable*, along with overall accuracy (Acc). N is the number of validation examples per language.

Language	Acc	Impossible			Answerable		
		P	R	F1	P	R	F1
Arabic (ar)	0.93	0.96	0.42	0.59	0.92	1.00	0.96
Korean (ko)	0.95	0.00	0.00	0.00	0.95	1.00	0.97
Telugu (te)	0.76	0.45	0.05	0.10	0.76	0.98	0.86

We see the results of the classification on table 6. This classifier rarely picks impossible, with picking impossible 24% of the time for Telugu, and 12.5% and 5% for Arabic and Korean respectively. F1, for both Korean and Telugu when answering Impossible, is very low, which leads us to believe that the high accuracies come from class imbalance. For Korean, the extreme skew (5% impossible) leads to mode collapse. Telugu classfies impossible the most, but the recall is very low which means that it mislabels answerable questions as impossible often. Arabic finds a meaningful portion of "impossible" (R=0.42) while keeping Answerable recall at 1.00. This is the only language where the SVM clearly beats the trivial baseline.

3.2 Logistic Regression classifier

A Logistic Regression classifier with TF–IDF features was trained as a probabilistic counterpart to the margin-based SVM. Questions and contexts were concatenated with a `[SEP]` token, and TF–IDF vectors (10k features, uni- and bigram range) were extracted using regex tokenization (`\S+`). Training used `LogisticRegression(max_iter=500)`.

This model offers calibrated probabilities, making it useful for interpretable confidence estimates. TF–IDF features capture both single keywords and short indicative phrases (e.g., "in which year") relevant to question answerability.

3.2.1 Evaluation

On the Arabic validation set ($N = 415$), the model achieved 0.88 accuracy. Table 7 shows detailed results.

Table 7: Logistic Regression (TF–IDF) results on Arabic validation set.

Class	**P**	**R**	**F1**
Impossible	1.00	0.08	0.14
Answerable	0.88	1.00	0.94
Accuracy		0.88	

3.2.2 Discussion

The classifier performs strongly on *Answerable* (F1 = 0.94) but struggles to recall the minority *Impossible* class (R = 0.08), despite perfect precision. High overall accuracy thus reflects class imbalance rather than balanced performance. Compared with the SVM, Logistic Regression yields similar trends—bias toward the majority class—but adds probabilistic outputs useful for integration into downstream systems. Addressing imbalance through class weighting or resampling would likely improve minority-class recall.

3.3 mBERT classifier

For our third classifier, we fine-tuned the multilingual BERT model as a binary classifier. The motivation for this choice is that mBERT provides contextualised subword representations across more than 100 languages, including Arabic, Korean and Telugu. We combined each question and its corresponding English context into a single input sequence and trained mBERT with a classification head predicting whether the question is answerable or impossible. The results demonstrate the strength of contextual multilingual encoders for this task. Arabic and Korean reach near-ceiling performance, with F1 scores above 0.98, indicating that mBERT can robustly separate answerable from impossible questions in these languages. Telugu, however, shows substantially lower accuracy (0.85) and F1 (0.91). The gap suggests that while mBERT benefits from shared multilingual representations, its coverage and pre-training quality are uneven across languages, with Telugu receiving less representation in pre-training compared to Arabic and Korean.

Language	Acc	P	R	F1
Arabic (ar)	0.981	0.994	0.983	0.989
Korean (ko)	0.975	0.982	0.991	0.987
Telugu (te)	0.849	0.852	0.969	0.907

4 Week 39

4.1 Generative QA for Telugu (Q(te)+C(en)->A(te))

We fine-tuned `google/mt5-base` to generate Telugu answers from Telugu questions and English contexts, using only samples with valid in-language answers.

Data. From `coastalcph/tydi_xor_rc`, Telugu items were cleaned by removing empty or placeholder answers in `answer_inlang`. After strict filtering, 50 training and 100 validation examples remained (from 1355/384 originally), ensuring high-quality supervision.

Model and training. The model was trained for 25 epochs (batch 2, LR 5×10^{-5}) with max lengths 512/64 and masked label padding. Decoding used 6-beam search with deterministic output.

Evaluation. Performance was measured with sacreBLEU on the validation split.

	Train N	Val N	Tokenizer src/tgt	BLEU (val)
mT5-base (ours)	50	100	512 / 64	8.25

Table 8: Week 39 Telugu generative QA. sacreBLEU on cleaned validation set.

Analysis. Despite the tiny dataset, the model learned basic alignment but failed on entity and number grounding, often generating generic or off-topic nouns. Longer English contexts and cross-lingual decoding make the task data-hungry. Larger beam width slightly improved recall but not BLEU.

Takeaways. (i) Data quality dominates performance—strict filtering prevents noise but leaves too few samples. (ii) Smaller models or longer training may stabilize learning in low-resource regimes. (iii) Augmenting missing Telugu answers via EN→TE translation could boost BLEU and enable finer per-class analysis.

4.2 mBERT Token Classification Model

We implemented a multilingual mBERT (`bert-base-multilingual-cased`) token classification model using the BIO tagging scheme (`O`, `B`, `I`) to predict answer spans in the English context for Arabic, Korean, and Telugu questions.

Language	P	R	F1
Arabic (ar)	0.625	0.597	0.611
Korean (ko)	0.739	0.536	0.621
Telugu (te)	0.545	0.430	0.481
Overall	0.642	0.533	0.582

The mBERT model achieves somewhat of a moderate performance, with an overall F1 score of 0.58. Korean achieved the highest F1, due to a higher precision while we have Telugu performing the worse due to a low recall. Arabic results were more or less balanced between these two metrics, which suggests that the model can identify relevant answer spans but still struggles with a portion of correct tokens which gives room for recall improvement. Some reasons for this performance can be linked to limited data, or differences in question morphology. Despite these errors/limitations, the model demonstrates that multilingual representations can indeed generalize across diverse languages (based on our 3) for span based QA.

5 Week 40

5.1 Span-based QA as Sequence Labelling (Telugu)

Task. Telugu QA is framed as token-level sequence labelling over the English context using the BIO scheme (`B`, `I`, `O`). Character-level spans (`answer_start`, `answer`) are aligned to tokens via `offset_mapping`, masking all non-context tokens.

Model and training. We fine-tuned `bert-base-multilingual-cased` for 3-way token classification ({O,B,I}) on (question, context) pairs with max length 384, batch 8, LR 3×10^{-5}, and 6 epochs. Unanswerable questions were trained as all-`O` sequences.

Evaluation. Token-level seqeval precision, recall, and F1 were computed on the Telugu validation set.

Language	Precision	Recall	F1	N (val)
Telugu (te)	0.334	0.399	0.364	384

Table 9: Span QA as BIO tagging over context tokens (Telugu).

Analysis. The model partially recovers short spans but often misaligns boundaries, especially

for multi-word entities and numbers. Most unanswerable cases are correctly predicted as all-O. Given cross-lingual inputs and limited data, performance is modest yet forms a strong baseline. Future work could test BIOES labelling, XLM-R initialization, or extended training for boundary refinement.

5.2 Question only generation model

For the open QA setting, we fine tuned `mT5-small` to make Telugu answers directly from the Telugu questions, without using the English context. Evaluation was performed with sacreBLEU.

Table 10: Question-only Telugu answer generation using mT5-small. BLEU score reported on the validation set.

Model	BLEU
mT5-small (question → Telugu answer)	0.058

The model shows limited/poor performance with the BLEU score at 0/058, producing short/generic outputs that may sometimes align semantically but do not often match the answers exactly. This is expected since we had a really small amount of Telugu data, and did not touch the English context. Still however, the model demonstrates partial knowledge and generates fluent Telugu text even when trained only on the question input.

5.3 English answer to in-language answer

For this section we used the google/mt5-small model, for the English to Telugu translation. The subset of the dataset lead to 50 training and 100 validation pairs.Training was run for 10 epochs with a learning rate of $5x10^{-5}$, batch size 8, and beam size 6 for generation.

Table 11: English answer to in-language answer generation using mT5-small. BLEU score reported on the validation set.

Model	BLEU
mT5-small (English answer→ Telugu answer)	0.08

Although the model occasionally produced valid transliterations, the small BLEU indicates minimal overlap with reference answers. A factor for this could be the data sparsity, which is especially relevant as the mT5-small model tends to default to sentinel tokens when exposed to extremely limited data.

6 Week 41: Evaluation on New Questions

6.1 Vietnamese model

We evaluated 4 models based on 10 Vietnamese questions, with English contexts: (1) rule based answerability heuristic (2) an mBERT answerability classifier (3) mBERT BIO over English context (span based sequence labeler) (4) mT5-small fine tuned to produce Telugu as a question only generator. The rule based baseline performs quite poorly

Model	Metric	Score
Rule-based (answerability)	Acc	0.20
mBERT cls (answerability)	Acc	0.90
Span labeler (BIO)	P / R / F1	0.684 / 0.900 / 0.726
mT5 (question-only)	BLEU	0.00

on Vietnamese. The mBERT classifier generalizes well across languages on this small set, and is helped by the multilingual encoder. The span labeler seems to achieve a high recall but falls short with precision due to boundary errors. Something to note is that the BLEU score of 0.0 is due to the fact that it is trained to output Telugu, while we have the references here being Vietnamese which is a language mismatch rather than a failure (or how the original model was programmed). Just as a sidenote – is this fine, or should there be a translator inputted prior to feeding the generator?

6.2 Evaluation on Custom Test Set

We created a custom Telugu test set (`testmahdi.json`) of ten question–context pairs with English passages. Each entry includes both English and Telugu answers, with verified `answer_start` indices. Eight questions are answerable and two are not. We evaluated four systems from prior weeks: a rule-based baseline (W36), TF–IDF + Logistic Regression (W38), mT5 generation (W39), and the span labeler (W40).

Findings. The rule-based method fails on this set (F1 = 0), confirming its reliance on surface overlap. The TF–IDF model performs reliably (F1 = 0.89) with perfect recall for answerable cases. The fine-tuned mT5 produces sentinel-like fragments (BLEU = 0), showing poor Telugu generation given limited fine-tuning data. The span model attains solid sequence F1 = 0.80, correctly predicting empty spans for unanswerable cases.

Model	Acc	P	R	F1
Rule-based	0.20	0.00	0.00	0.00
TF–IDF + LogReg	0.80	0.80	1.00	0.89
Span labeler (BIO)	–	0.86	0.75	0.80

Generative model	BLEU
mT5 (fine-tuned)	0.00

Table 12: Week 41 results on Telugu test set (N=10; 8 answerable / 2 unanswerable).

Overall, extractive models generalize better than generative ones, and trained classifiers outperform heuristic baselines.

7 Contributions

Week 36

Mahdi Mohammadzadeh: Translation to English for baseline; main report writing; expanded discussion.
Jonasz Witczak: Rule-based classifier design and implementation.
Hung Pham: Dataset statistics; frequent-question-word analysis; tables/figures implementations.

Week 37

Hung Pham: Unigram language model implementation; training/evaluation; perplexity reporting, and writing of own implementation
Mahdi Mohammadzadeh: Bigram and trigram language model implementation; training and validation on all languages; results integration into tables; explanation and analysis in the report.
Jonasz Witczak: Bigram and trigram reporting.

Week 38

Jonasz Witczak: Linear SVM implementation; Reporting of Linear SVM; Traning and evaluation on all 3 language validatio n sets
Mahdi Mohammadzadeh: Logistic Regression (TF–IDF) implementation; training and evaluation on Arabic validation set; results integration and detailed analysis in the report.

Hung Pham: mBERT classifier implementation evaluation on Arabic, Korean, and Telugu validation sets; integration of results and analysis into the report.

Week 39

Mahdi Mohammadzadeh: Telugu generative QA (mT5) pipeline: data cleaning of `answer_inlang` (strict filtering), model fine-tuning, deterministic decoding and sacreBLEU evaluation, qualitative error analysis, and write-up integration.
Hung Pham: Implemented mBERT token classification model (BIO tagging) for span-based QA; trained and evaluated on Arabic, Korean, and Telugu; prepared results and table for the report, did analysis.

Week 40

Mahdi Mohammadzadeh: Telugu span-based QA (BIO) — char→token alignment from `answer_start`/`answer`; mBERT fine-tuning; seqeval (P/R/F1) evaluation; error analysis; report write-up and table integration.
Hung Pham: 5.2 section, question only generation model implementation, analysis, and report.
Jonasz Witczak: 5.3 section, answer only generation model implementation, analysis, and report.

Week 41

Hung Pham: 6.1 section, Vietnamese model evaluation implementation, analysis, and report.
Mahdi Mohammadzadeh: Created Telugu test set; ran evaluations for all four models (rule-based, TF–IDF, mT5, span labeler); compiled and analysed final results.