PolQA: Polish Question Answering Dataset

Piotr Rybak[1], **Piotr Przybyła**[1,2], **Maciej Ogrodniczuk**[1]
[1] Institute of Computer Science, Polish Academy of Sciences
ul. Jana Kazimierza 5, 01-248 Warsaw, Poland
[2] TALN Group, Universitat Pompeu Fabra
c. Tànger 122-140, 08018 Barcelona, Spain
{firstname.lastname}@ipipan.waw.pl

Abstract

Recently proposed systems for open-domain question answering (OpenQA) require large amounts of training data to achieve state-of-the-art performance. However, data annotation is known to be time-consuming and therefore expensive to acquire. As a result, the appropriate datasets are available only for a handful of languages (mainly English and Chinese). In this work, we introduce and publicly release PolQA, the first Polish dataset for OpenQA. It consists of 7,000 questions, 87,525 manually labeled evidence passages, and a corpus of over 7,097,322 candidate passages. Each question is classified according to its formulation, type, as well as entity type of the answer. This resource allows us to evaluate the impact of different annotation choices on the performance of the QA system and propose an efficient annotation strategy that increases the passage retrieval accuracy@10 by 10.55 p.p. while reducing the annotation cost by 82%.

Keywords: Polish, Open-domain Question Answering

1. Introduction

The goal of open-domain question answering (OpenQA) is to provide an answer to a question asked in a natural language. Typically, an OpenQA system consists of three components. The *knowledge source* is used as a source of passages that might contain the answer. The *retriever* is responsible for searching for relevant passages from the knowledge source. Finally, the *reader* extracts (or generates) the answer based on the given question and retrieved passages.

Recently, neural retrieval systems (e.g. Dense Passage Retrieval, Karpukhin et al., 2020, DPR) surpassed traditionally used lexical methods (e.g. BM-25, Robertson and Zaragoza, 2009) by fine-tuning pre-trained language models on a large number of (question, passage) pairs. They achieve state-of-the-art results but at the cost of the necessity to annotate training sets and poor generalizability to other languages or even domains (Thakur et al., 2021).

The third component of OpenQA systems (*reader*) also requires annotated dataset for training. Besides (question, passage) pairs, it needs the final answer to the question. Reader takes the form of either a short span of the passage (extractive QA) or free text (generative QA).

In this paper, we introduce and release PolQA, the first Polish OpenQA dataset. It consists of 7,000 general knowledge questions obtained from TV quiz shows, online quizzes, and similar sources. Each question is accompanied by up to 15 evidence passages (87,525 in total) and up to five answer variants (8,713 in total). We also release a corpus of 7,097,322 candidate passages based on parsed Polish Wikipedia. Additionally, each question is described by its formulation (how the question is asked), type (what kind of information is sought), and the entity type of the answer (e.g. country, person).

The wide availability of pretrained language models means that a decently-performing system may be built with little to no training data, but adding some annotation will result in much better results. Thus, a system designer needs to predict, adding what (and how much) manually annotated data will be most cost-effective. During the course of creating PolQA, we analyzed the impact of several annotation strategies on the QA system performance and annotation cost. This allows us to answer the following research questions:

- **RQ1**: What is the benefit of high-quality and unbiased training data?
- **RQ2**: Does the performance improve with more training examples?
- **RQ3**: What is the impact of manually annotated hard negative passages on different components of the OpenQA system?
- **RQ4**: What is the cost of obtaining annotations in terms of human effort?
- **RQ5**: What annotation strategy can be recommended for future OpenQA annotation efforts based on the above?

To summarize, our contributions are:

LREC-COLING 2024, pages 12846–12855
20-25 May, 2024. © 2024 ELRA Language Resource Association: CC BY-NC 4.0

1. Release of PolQA: the first Polish OpenQA dataset,[1]

2. Empirical study of data annotation strategies for OpenQA and proposal of an efficient method to create similar datasets for other languages.

2. Related Work

OpenQA is an established task in natural language processing research (Zhu et al., 2021). Over the years, multiple datasets were published and used for training and evaluation of OpenQA systems.

The first version of MS MARCO dataset (Nguyen et al., 2016) consisted of 100,000 questions sampled from Bing's search logs and matching passages obtained from top 10 Bing's search results. Since then the dataset was updated a few times and currently has 1,010,916 questions. Similar to MS MARCO, the contributors of Natural Questions (Kwiatkowski et al., 2019, NQ) sampled 323,045 questions from Google search logs. However, they limited the possible passages to Wikipedia articles. Another OpenQA dataset created using search logs is DuReader (He et al., 2018). In contrast to MS MARCO and NQ, DuReader is in Chinese. It makes it the largest (and one of few) OpenQA dataset for the non-English language.

The QA datasets for Polish are more scarce. *Czy wiesz?* dataset (Marcińczuk et al., 2013) consists of 4,721 questions from the *Did you know?* section on Polish Wikipedia. However, only 250 of them were manually matched with a relevant passage, the rest was only matched with the whole article. Rybak et al. (2020) extended the aforementioned dataset by labeling passages for additional 1,070 questions. The lack of answers limits the usability of the dataset to training only passage retriever models. The OpenQA system RAFAEL (Przybyła, 2016) used 1,130 questions collected from a Polish quiz TV show *Jeden z dziesięciu* (Karzewski, 1997). The dataset contains answers, which allow end-to-end evaluation of the OpenQA system, but lacks question-passage pairs. Two recent resources are related to the PolEval 2021 shared task on QA (Ogrodniczuk and Przybyła, 2021). The official dataset contains 6,000 questions together with matching answers. Moreover, one of the participating teams gathered additional 1,000 question-answer pairs (Rybak, 2021). Again, both datasets lack matching passages. The latest Polish dataset is PoQuAD (Tuora et al., 2022, 2023), a Polish equivalent of SQuAD (Rajpurkar et al., 2018). It consists of 70,000 question-answer pairs with passages extracted from Polish Wikipedia. Unlike the PolQA dataset, it was created by asking annotators to write a question about a given passage, rather than by finding the passage to a given question. Often these questions are only valid in the context of the given passage, e.g. "What day did the battle start?", which makes them suitable for reading comprehension but not for neural retrieval.

Few multilingual resources include Polish text. The Cross-lingual OpenQA dataset (Liu et al., 2019, XQA) consists of an English training set and evaluation data for additional eight languages (including 1,846 questions for Polish). Similar to the *Czy wiesz?* dataset, it was created from the *Did you know?* section of Wikipedia. However, the resulting dataset consists of cloze test statements instead of grammatically correct questions. The Multilingual Knowledge Questions & Answers (Longpre et al., 2020, MKQA) contains 10,000 questions sampled from NQ and manually translated into 25 typologically diverse languages. As noted by Rybak (2021), over 80% of those questions are not useful for training the OpenQA model as they lack the answer, are ambiguous, or require a long answer (*Why?* and *How?* questions). None of the above resources have matching passages.

3. Data Collection

Typical architecture of an open-domain QA system consists of two models – retriever and reader (Zhu et al., 2021). Retriever finds passages from the corpus that might be relevant to the question. Reader uses those passages to extract (or generate) the final answer. In this section, we describe our approach to annotate triplets of (question, passage, and answer) required to train such systems.

The annotation team consisted of 16 annotators, all native Polish speakers, most of them having linguistic backgrounds and previous experience as annotators. The authors of this study acted as super-annotators, who kept the annotators' work in line with the guidelines. In particular, they reviewed the first 200 labeled examples of each annotator to provide feedback on their work and improve the quality of further annotations. In addition, the super-annotators provided ongoing support during the annotation process, helping with ambiguous examples and clarifying any doubts about the guidelines.

3.1. Questions and Answers

The majority of questions come from two existing resources, the 6,000 questions released during the PolEval 2021 shared task on QA (Ogrodniczuk and Przybyła, 2021) and additional 1,000 questions gathered by one of the shared task participants (Rybak, 2021). Originally, the questions come from collections associated with TV shows, both officially published (Karzewski, 1997) and gathered online by their fans, as well as questions used in actual quiz competitions, on TV or online.

[1] PolQA dataset is available at: `https://hf.co/datasets/ipipan/polqa`

Answers are formulated in a natural language, in a way a Polish speaker would answer the questions. It means that the answers might contain prepositions, be inflected, and contain punctuation. In some cases, the answer might have multiple correct variants, e.g. numbers are written as numerals and words, synonyms, abbreviations and their expansions. We include all such variants.

During the annotation, we cleaned the existing dataset by correcting the factual correctness of questions, adding missing answer variants, and replacing near-duplicates with new questions.

3.2. Taxonomy

We manually classify each question-answer pair based on its (1) formulation, (2) question type, and (3) entity type, according to the taxonomy proposed by Ogrodniczuk and Przybyła (2021).

Formulation denotes the kind of expression used to request information. Three types of phrasing are distinguished:[2]

- **plain question**, e.g. *What is the name of the first letter of the Greek alphabet?*
- **command**, e.g. *Expand the abbrev. 'CIA'.*
- **compound**, e.g. *This French writer, born in the 19th century, is considered a pioneer of the sci-fi literature. What is his name?*

Question type indicates what type of information is sought by the question:

- **single entity**, e.g. *Who is the hero in the Tomb Raider video game series?*,
- **multiple entities**, e.g. *Which two seas are linked by the Corinth Canal?*,
- **entity choice**, e.g. *Is 'Sombrero' a type of a dance, a hat or a dish?*,
- **yes/no**, e.g. *When the term of office of the Polish Sejm is terminated, does it apply to the Senate as well?*,
- **other entity name**, e.g. *What was the nickname of Louis I, the King of the Franks?*,
- **gap filling**, e.g. *Finish the proverb: 'if you fly with the crows...'*.

The question regarding entities can either seek a **named entity** or an **unnamed one**. In the former case, the questions are categorised according to the fine-grained **named entity type**.[3]

3.3. Source of Passages

We chose Wikipedia as our source of passages as it contains relevant passages for over 93% of all questions. The missing questions concern basic arithmetic, proverbs, the content of books or movies, are yes/no questions with a negative answer, or comparison questions, which require multiple passages to answer.

3.4. Candidate Passages

Each Wikipedia article is parsed using WikiExtractor (Attardi, 2015), but contrary to default settings we keep lists as a valid text. We split parsed articles into passages at the ends of the paragraphs or if the passage is longer than 500 characters. We try to split on sentence boundaries, whenever possible. Overall, we obtain a knowledge source of 7,097,322 passages.

3.5. Evidence Passages

We use binary relevance score to annotate passages in the context of asked question. We define a relevant (also called *positive*) passage as a continuous span sentences which allows to answer a question assuming basic reasoning skills (e.g. the conversion of years to centuries) and knowledge (e.g. Poland is a country). Otherwise, the passages is considered irrelevant (or *negative*).

The process of selecting negative passages is crucial for the final performance of neural retrievers. Usually, the best results are achieved when the negative passages are very similar to the given question, i.e. it is hard to decide whether the passage is positive or negative. Therefore, such negatives are often called *hard* negatives. Although there are many methods to select them automatically (Ren et al., 2021; Karpukhin et al., 2020), we decided to evaluate if it is beneficial to label them manually.

3.6. Annotation Strategies

We follow two strategies to annotate passages for each question and named them *standard* and *efficient*. Each of them consists of two phases: the retrieval of candidate passages and the manual verification of their relevance. In particular, the verification phase is the source of hard negative passages, since passages considered positive in the retrieval phase can be labeled negative in the verification phase.

Standard strategy The first strategy is to ask annotators to use internal (i.e. Wikipedia Search) or external (e.g. Google) search engines to find a relevant passage in the knowledge source using any keywords or queries they consider useful.

[2] The examples are translated into English for the convenience of the reader.

[3] We include 34 types: day, year, century, period, count, quantity, person, name, surname, dynasty, organisation, company, band, country, state, city, nationality, mountain, lake, island, sea, river, range, archipelago, continent, place, vehicle, title, symbol, event, celestial body, animal, building, and other.

Strategy	# Questions			# Passages		
	Positive	Negative	Total	Positive	Negative	Total
Standard	6,427	6,886	7,000	29,841	28,991	58,832
Manual retrieval	6,296	1,763	7,000	21,451	2,729	24,180
Neural retrieval	4,456	6,868	7,000	8,714	26,286	35,000
Efficient	5,402	6,505	7,000	13,191	21,809	35,000
Total	6,516	6,946	7,000	38,908	48,617	87,525

Table 1: Number of questions and evidence passages in the PolQA dataset for each annotation strategy. The passage is positive (negative) if it contains (does not contain) an evidence supporting answer. For questions, it refers to the number of questions with at least one positive/negative passage. The number of passages does not sum up because some passages were present in more than one annotation strategy.

We hypothesize that the unconstrained way of finding the passages will result in more unbiased and diverse examples. Moreover, we ask the annotators to find not one, but up to five passages, preferably from different articles to even further increase passage diversity.

To assure the correctness of the found passages, during the verification phase we double-check each of them to decide if they are relevant (i.e. allow to answer the question) or not. This is the first source of hard negative passages for this strategy. However, since annotators were tasked with finding only relevant passages the number of negatives is relatively small (around 11% of all labeled passages, see Table 1).

To overcome the scarcity of negative passages, we train a neural retriever on the aforementioned passages (see row 3 from Table 5) and use it to retrieve additional 5 most relevant passages for each question. Then, the annotators manually verified each passage to decide if it is relevant or not. We use irrelevant passages as the second source of hard negatives.

Efficient strategy An alternative approach to annotating passages is to show annotators question-passage pairs and ask them to verify if the passage is relevant or not. This method is several times faster (see Table 4), but it requires a sampling function that selects passages to annotate. Choosing the wrong function can lead to inefficiency (if it selects irrelevant passages) and might bias the dataset (e.g. by selecting passages with high lexical overlap).

We propose the following pipeline as a sampling function. First, we use SpaCy (Honnibal et al., 2020) to lemmatize questions and passages. Then, the BM-25 algorithm (Robertson and Zaragoza, 2009) selects top 100 candidate passages which we re-rank using multilingual cross-encoder (Bonifacio et al., 2021).[4] Finally, we select top 5 passages per questions for manual verification.

We purposefully avoid any Polish-specific resources to assure the generality of our approach. Only the lemmatiser is trained on Polish texts, however, it is not required for the method to work and lemmatisers are widely available for many languages (Qi et al., 2020).

4. PolQA Dataset

4.1. Data Statistics

The final PolQA dataset consists of 7,000 questions, 8,713 answer variants, 87,525 evidence passages, and a corpus of 7,097,322 unlabeled candidate passages. For each question, we used two different annotation strategies (see Section 3.5) to obtain a diverse set of evidence passages. Each strategy results in a different ratio of positive and negative passages (see Table 1). By design, the *Standard*$_{manual}$ strategy has the highest share of positive passages. Any negatives come from the quality assessment phase during which 11% of passages were considered not to answer the question (see Table 4). Both *Standard*$_{neural}$ and *Efficient* strategy have much more negatives than positives. Overall, 93% of questions have at least one positive passage and 99% have at least one negative passage.

4.2. Question Types

Each question is classified according to three different dimensions (see Section 3.2). Plain questions account for 97.7% of cases (see Table 2). There are 1.5% of compounds, usually, a statement with an introduction followed by a question. The rest 0.8% of the questions are commands.

Most questions ask for a single entity (79.8%) or let to choose a single entity among a few provided (10.1%). There are 7.6% yes/no questions, and a small share of other question types.

[4] `https://hf.co/unicamp-dl/mMiniLM-L6-v2-mmarco-v2`

Category	Questions	Share
Question formulation		
Plain question	6,839	97.7%
Compound	108	1.5%
Command	53	0.8%
Question type		
Single entity	5,589	79.8%
Entity choice	705	10.1%
Yes/No	532	7.6%
Other name	86	1.2%
Multiple entities	59	0.8%
Gap filling	29	0.4%
Entity type		
Named	3,854	55.1%
Unnamed	2,614	37.3%
Yes/No	532	7.6%

Table 2: Distribution of questions based on their formulation, type, or type of entity.

Strategy	Token overlap		Lemma overlap	
	Pos	Neg	Pos	Neg
Questions				
Standard				
Manual	15.5%	12.5%	35.6%	28.7%
Neural	20.3%	13.1%	42.8%	27.8%
Efficient	25.1%	23.1%	51.5%	47.9%
Answers				
Standard				
Manual	41.1%	33.8%	71.5%	60.8%
Neural	40.1%	10.4%	70.2%	16.3%
Efficient	45.0%	14.5%	75.4%	22.9%

Table 3: Average token/lemma overlap between questions/answers and positive (*Pos*) or negative (*Neg*) passages. We calculate token/lemma overlap as a percentage of question/answer tokens/lemmas (excluding punctuation) that appear in the matching passage. We take a maximum if there is more than one answer. We exclude questions with *yes/no* answers which might have only accidental overlap.

There is greater variety in entity type, the 55.1% answers being named entities and 37.3% of unnamed entities.

4.3. Lexical Similarity

One of the main limitations of the traditional retrieval algorithms (e.g. BM-25) is their dependence on the lexical overlap. If there is no common token between the question and the relevant passage then it won't be found by the retriever. However, the OpenQA datasets are often created by automatically finding the candidate passages first and then asking annotators to assess if they are positive or not (see Section 2). This might introduce bias in the final dataset as the method of finding candidate passages might rely on the lexical overlap. As a result, the lexical methods will perform well on such a dataset.

We analyze the similarity between questions and passages by calculating both token (before lemmatisation) and lemma overlap as a percentage of question tokens/lemmas (excluding punctuation) that appear in the matching passage (see Table 3). As expected, the *Standard*$_{manual}$ strategy of finding passages has a much lower lemma overlap for positive passages (35.6%) than *Efficient* method which used a lexical retriever to find candidate passages (51.5%). Interestingly, even the negative passages obtained with *Efficient* method have higher overlap (47.9%) which makes them perfect as hard negatives for training a neural retriever. The *Standard*$_{neural}$ strategy sits in between those two methods, the usage of neural retriever leads to higher overlap for positive methods (42.8%) and comparable for negative ones (27.8%).

The same type of analysis is beneficial for comparing passages and the answers. We calculate both token and lemma overlap to understand how often the answer is directly present in the passage (see Table 3) and can be simply extracted by the reader and when it has to be generated. We excluded the yes/no questions since in those cases any overlap would be accidental and meaningless. The annotation strategy has a minimal influence on the overlap for positive passages with *Efficient* method having the highest overlap and *Standard*$_{neural}$ the lowest. For *Standard*$_{manual}$ strategy, the overlap between positive and negative passages is similar since annotators were asked to only find positive passages. For other methods, the negative passages have much lower overlap, i.e. do not contain answers. The token overlap of around 40% indicates the high difficulty of the PolQA dataset. The reader cannot fall back on copying the correct span of the passage but has to generate an answer containing novel tokens. Partially, the difficulty is solved by learning how to lemmatise but even the lemma overlap is still only around 70%.

Task	Time	Precision	Recall
Free search	75.5	88.7%	96.6%
Verification	13.6	89.9%	95.8%

Table 4: Data annotation statistics depending on the type of the annotation task, freely searching for a passage or verifying if a given passage is correct. Time refers to the average time in seconds to annotate one question-passage pair.

4.4. Annotation Quality

There is a significant time difference between freely searching for a passage (*Standard$_{manual}$* method) and verifying if the passage is correct (*Efficient* and *Standard$_{neural}$* methods). On average, searching for the passage takes 75.5 seconds per passage while verifying one is over five times faster and takes only 13.6 seconds (see Table 4). There is no significant difference between the correctness of passages obtained by both methods with a precision of around 89% and a recall of around 96%.

5. Evaluation

We use standard retriever-reader architecture to train an OpenQA system and evaluate different annotation strategies. We split the dataset into the train (5,000 questions, 27,131 relevant passages), validation (1,000 questions, 5,839 relevant passages), and test (1,000 questions, 5,938 relevant passages) sets. To avoid potential bias introduced by the annotation approach, we limit the validation and test sets to passages found using *Standard$_{manual}$* method (3,160 and 3,252 relevant passages respectively). The experiments use the validation set to evaluate the models, except for the final evaluation (Section 6.5), where the test set is employed. We use a single V100 GPU for all experiments.

5.1. Retriever

As a retriever, we use HerBERT Base model (Mroczkowski et al., 2021) and fine-tune it with a triplet loss (Weinberger and Saul, 2009) and a margin of 0.1. We train this encoder with AdamW optimizer (Loshchilov and Hutter, 2019) for 50 epochs using a learning rate of 10^{-5} and a linear decay schedule. During the training, we sample one positive and one negative passage for each question. We randomly sample negatives from a training set, except for the experiments in Section 6.4, where manually labeled negative passages are used instead. In all cases, we also use in-batch negatives. During the evaluation, we first encode all Wikipedia passages and index them using FAISS (Johnson et al., 2019). Then, for each question, we retrieve the top 10 most similar passages through an exhaustive search. We measure model performance through the accuracy of top 10 candidates.

5.2. Reader

As a reader, we use plT5 Base model (Chrabrowa et al., 2022) and train it in a text-to-text mode. We concatenate the question with all available relevant[5] passages as input and generate a sequence of tokens as an answer. We fine-tune the model with AdaFactor optimizer (Shazeer and Stern, 2018) for 10 epochs using a learning rate of 10^{-4} and a linear decay schedule.

To evaluate the reader (as well as the end-to-end system), we use the metric proposed by Ogrodniczuk and Przybyła (2021). For numerical answers, we extract the numeral (Arabic or Roman) using regular expression and expect the equality between prediction and true value. For the rest of the questions, we calculate character-wise Levenshtein distance (Levenshtein, 1966), which is allowed to reach 50% of the answer length for a match. In case there is more than one correct answer, we compare the prediction to each and choose the best matching ones.

We evaluate the reader both on manually labeled positive passages (to assess the reader quality itself), as well as on passages returned by the retriever for an end-to-end evaluation.

6. Results

To understand how different annotation strategies influence the performance of an OpenQA system, we conduct a series of experiments to answer the aforementioned research questions (see Section 1):

- In Section 6.1 and 6.3, we investigate the benefit of high-quality and unbiased training data (RQ1).
- In Section 6.2, we analyze how the performance improves with the increased number of training passages (RQ2).
- In Section 6.4, we compare models with and without the annotated hard negative passages (RQ3).
- In Section 6.5, we summarize our experiments and compare two annotation strategies based on their cost (in terms of human effort) and impact on OpenQA performance (RQ4, RQ5).

6.1. Quality Assessment

We train a model using all passages from *Standard$_{manual}$* strategy and compare it to the model

[5] Except for the experiments in Section 6.4, where manually labeled negative passages are also used.

#	Strategy	Verified	# Positives	# Hard Negatives	Retriever	Reader	E2E
1	Standard	No	Single	No	52.57%	75.33%	46.54%
2	Standard	Yes	Single	No	52.90%	72.99%	44.42%
3	Standard	Yes	All	No	53.24%	80.25%	47.77%
4	Standard	Yes	All	Yes	46.76%	**80.92%**	51.23%
5	Efficient	No	All	No	54.24%	77.68%	51.23%
6	Efficient	Yes	All	No	59.26%	79.46%	52.46%
7	Efficient	Yes	All	Yes	**61.16%**	77.79%	**56.25%**

Table 5: OpenQA model performance trained on passages obtained using different annotation strategies. We use the top 10 accuracy on the validation set. Verified refers to whether the passage was additionally verified by the annotator or taken as-is. Positives refers to the number of passages per question. Hard Negatives refers to whether hard negatives (passages manually verified as negatives) were used in training.

using only verified ones. Even though the precision of human passages is around 89% (see Table 4) the impact of additional verification on retriever performance is negligible (see rows 1, 2 in Table 5). Using only verified passages actually decreases the accuracy of the reader by 2.34 p.p. It has a similar impact on the end-to-end score (a decrease from 46.54% to 44.42%).

6.2. Number of Relevant Passages

The next design choice in the *Standard$_{manual}$* strategy is how many relevant passages should be found for each question. Increasing this number heavily influences the annotators' workload, since every next passage for a given question usually takes more time to find. We compare models trained with a single or all (i.e. up to five) passages per question (see rows 2, 3 in Table 5). In both cases, we use only verified passages. Again, the difference for the retriever is positive but small (52.9% vs 53.24%). For the reader, the number of used passages is more important. The model trained on all passages scores 7.25 p.p. higher compared to one trained on only one passage per question. The impact of the number of passages on the end-to-end accuracy is also positive as it increases the performance by 3.35 p.p.

6.3. Retrieval vs Verification

We compare two main strategies to annotate the passages, freely searching for relevant passages (*Standard$_{manual}$*) or manual verification of passage candidates (*Efficient*). Additionally, we train a model on passages from *Efficient* but without any manual annotation, i.e. treating all passages returned by the sampling function as positives. This can be viewed as a simple model distillation (Ren et al., 2021).

Surprisingly, the retriever trained on unlabeled examples obtained using *Efficient* strategy performs better than the retriever trained on manually annotated data (see rows 3, 5 in Table 5). It shows that thanks to the high generalizability of cross-encoders it is possible to use a multilingual model to automatically find relevant passages and use them to create a high-quality dataset.

If we additionally verify examples obtained with *Efficient* strategy, we get an additional 5 p.p. improvement in retriever performance (see rows 5, 6 in Table 5). The manual annotation increases also the accuracy of the reader (from 77.68% to 79.46%) and the whole system (from 51.23% to 52.46%). However, the best reader performance (80.25%) is achieved by training on *Standard$_{manual}$* passages.

6.4. Hard Negatives

We explore the impact of hard negatives on model performance by using manually annotated negative passages instead of randomly sampled ones. For *Standard* strategy, we additionally use all passages obtained with *Standard$_{neural}$* method (see Section 3.5). For *Efficient* we simply include the passages annotated as negatives.

Including hard negatives in *Standard* dataset decreases the retriever performance from 53.24% to 46.76% (see rows 3, 4 in Table 5). The opposite happens for the reader. Additional passages slightly improve the accuracy of the reader (by 0.67 p.p.) which leads to an end-to-end increase of 3.46 p.p.

For the *Efficient* method, the hard negatives improve the retriever performance by almost 2 p.p. and at the same time hurt the reader performance by almost 2 p.p. However, the end-to-end accuracy increases from 52.46% to 56.25% (see rows 6, 7 in Table 5).

Strategy	Retriever	Reader	E2E	Time
Standard	51.47%	78.45%	51.47%	376
Efficient	62.02%	74.43%	53.21%	68

Table 6: OpenQA model performance trained on passages obtained using standard and efficient annotation strategy. We use the top 10 accuracy on the test set. Time refers to the average time in seconds to annotate passages for one question.

6.5. Summary

To summarize, we compare two data annotation strategies. In *Standard* approach, we use all manually verified *Standard*$_{manual}$ passages for training retriever. For reader, we additionally include all *Standard*$_{neural}$ passages. The *Efficient* approach uses verified passages (both positives and negatives) from *Efficient* method.

For retrievers, the model trained on passages obtained with *Efficient* strategy results in 10 p.p. higher accuracy compared to *Standard* approach (see Table 6). For readers, the *Standard* approach works better by almost 4 p.p. The end-to-end for both methods are similar, the *Efficient* method has an accuracy of 53.21% compared to 51.47% for *Standard* annotation strategy. Although the final results are similar, the time spent annotating the data is very different. The *Efficient* approach requires over five times less time to annotate passages for a single question (68 vs 376 seconds).

7. Conclusion

In this work, we present PolQA, the first Polish dataset for OpenQA. It consists of 7,000 questions together with 8,713 answer variants and 87,525 evidence passages obtained by different methods to increase their diversity and completeness.

This resource allows us to evaluate the performance of the OpenQA model depending on different data annotation strategies and formulate the following recommendations for creating a similar OpenQA dataset for other languages:

- Obtaining unbiased evidence passages does not improve the performance of OpenQA models. Instead, we recommend using *Efficient* strategy to sample candidate passages and manually verify their correctness. This reduces the cost of annotation over five times and at the same time increases the performance of the OpenQA model (see Section 6.3).

- If the annotation cost is still a limiting factor, then using unlabeled passages retrieved by the sampling function from *Efficient* strategy is a competitive (or even better) strategy than obtaining unbiased passages but requires no manual annotation (see Section 6.3).

- It is beneficial to annotate multiple evidence passages per question, as well as to include not only positive but also negative passages (see Section 6.2 and 6.4).

- Depending on the experience and skill of the annotators, it might not be necessary to double-check their work (see Section 6.1).

We hope our work will enable research on Polish OpenQA and be beneficial to the wider OpenQA research community, both to researchers working on cross-lingual OpenQA and those who seek an efficient approach to create OpenQA datasets.

8. Acknowledgments

This work was supported by the European Regional Development Fund as a part of 2014–2020 Smart Growth Operational Programme, CLARIN — Common Language Resources and Technology Infrastructure, project no. POIR.04.02.00-00C002/19.

The work was financed as part of the investment: "CLARIN ERIC – European Research Infrastructure Consortium: Common Language Resources and Technology Infrastructure (period: 2024-2026) funded by the Polish Ministry of Science and Higher Education (Programme: "Support for the participation of Polish scientific teams in international research infrastructure projects"), agreement number 2024/WK/01.

9. Bibliographical References

Giuseppe Attardi. 2015. Wikiextractor. https://github.com/attardi/wikiextractor.

Luiz Henrique Bonifacio, Vitor Jeronymo, Hugo Queiroz Abonizio, Israel Campiotti, Marzieh Fadaee, , Roberto Lotufo, and Rodrigo Nogueira. 2021. mmarco: A multilingual version of ms marco passage ranking dataset.

Aleksandra Chrabrowa, Łukasz Dragan, Karol Grzegorczyk, Dariusz Kajtoch, Mikołaj Koszowski, Robert Mroczkowski, and Piotr Rybak. 2022. Evaluation of Transfer Learning for Polish with a Text-to-Text Model. *arXiv:2205.08808*.

Wei He, Kai Liu, Jing Liu, Yajuan Lyu, Shiqi Zhao, Xinyan Xiao, Yuan Liu, Yizhong Wang, Hua Wu, Qiaoqiao She, Xuan Liu, Tian Wu, and Haifeng Wang. 2018. DuReader: a Chinese machine reading comprehension dataset from real-world applications. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 37–46, Melbourne, Australia. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. 10.5281/zenodo.1212303.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Kalervo Järvelin and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of ir techniques. *ACM Trans. Inf. Syst.*, 20:422–446.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Marek Karzewski. 1997. *Jeden z dziesięciu — pytania i odpowiedzi*. Muza SA.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Vladimir Iosifovich Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10:707–710.

Jiahua Liu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2019. XQA: A cross-lingual open-domain question answering dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2358–2368, Florence, Italy. Association for Computational Linguistics.

Shayne Longpre, Yi Lu, and Joachim Daiber. 2020. MKQA: A Linguistically Diverse Benchmark for Multilingual Open Domain Question Answering. *arXiv:2007.15207*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations (ICLR 2019)*.

Michał Marcińczuk, Adam Radziszewski, Maciej Piasecki, Dominik Piasecki, and Marcin Ptak. 2013. Evaluation of baseline information retrieval for Polish open-domain question answering system. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP 2013*, pages 428–435, Hissar, Bulgaria. INCOMA Ltd. Shoumen, BULGARIA.

Robert Mroczkowski, Piotr Rybak, Alina Wróblewska, and Ireneusz Gawlik. 2021. HerBERT: Efficiently pretrained transformer-based language model for Polish. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 1–10, Kiyv, Ukraine. Association for Computational Linguistics.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset.

Maciej Ogrodniczuk and Łukasz Kobyliński, editors. 2021. *Proceedings of the PolEval 2021 Workshop*. Institute of Computer Science, Polish Academy of Sciences, Warsaw.

Maciej Ogrodniczuk and Piotr Przybyła. 2021. PolEval 2021 Task 4: Question Answering Challenge. In (Ogrodniczuk and Kobyliński, 2021), pages 123–136.

Piotr Przybyła. 2016. Boosting Question Answering by Deep Entity Recognition. *arXiv:1605.08675*.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108, Online. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, QiaoQiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. 2021. RocketQAv2: A joint training method for dense passage retrieval and passage re-ranking. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: Bm25 and beyond. *Found. Trends Inf. Retr.*, 3:333–389.

Piotr Rybak. 2021. Retrieve and Refine System for Polish Question Answering. In (Ogrodniczuk and Kobyliński, 2021), pages 151–157.

Piotr Rybak. 2023. MAUPQA: Massive automatically-created Polish question answering dataset. In *Proceedings of the 9th Workshop on Slavic Natural Language Processing 2023 (SlavicNLP 2023)*, pages 11–16, Dubrovnik, Croatia. Association for Computational Linguistics.

Piotr Rybak, Robert Mroczkowski, Janusz Tracz, and Ireneusz Gawlik. 2020. KLEJ: Comprehensive benchmark for Polish language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1191–1201, Online. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

Ryszard Tuora, Natalia Zawadzka-Paluektau, Cezary Klamra, Aleksandra Zwierzchowska, and Łukasz Kobyliński. 2022. Towards a polish question answering dataset (poquad). In *From Born-Physical to Born-Virtual: Augmenting Intelligence in Digital Libraries: 24th International Conference on Asian Digital Libraries, ICADL 2022, Hanoi, Vietnam, November 30 – December 2, 2022, Proceedings*, page 194–203, Berlin, Heidelberg. Springer-Verlag.

Ryszard Tuora, Aleksandra Zwierzchowska, Natalia Zawadzka-Paluektau, Cezary Klamra, and Łukasz Kobyliński. 2023. Poquad - the polish question answering dataset - description and analysis. In *Proceedings of the 12th Knowledge Capture Conference 2023*, K-CAP '23, page 105–113, New York, NY, USA. Association for Computing Machinery.

Kilian Q Weinberger and Lawrence K Saul. 2009. Distance Metric Learning for Large Margin Nearest Neighbor Classification. *Journal of Machine Learning Research*, 10(2):207–244.

Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. 2021. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 1503–1512, New York, NY, USA. Association for Computing Machinery.

Fengbin Zhu, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua. 2021. Retrieving and reading: A comprehensive survey on open-domain question answering. *arXiv:2101.00774*.