

Harry Potter and the Action Prediction Challenge from Natural Language

David Vilares

Universidade da Coruña, CITIC
Departamento de Computación
Campus de Elviña s/n, 15071
A Coruña, Spain
david.vilares@udc.es

Carlos Gómez-Rodríguez

Universidade da Coruña, CITIC
Departamento de Computación
Campus de Elviña s/n, 15071
A Coruña, Spain
carlos.gomez@udc.es

Abstract

We explore the challenge of action prediction from textual descriptions of scenes, a testbed to approximate whether text inference can be used to predict upcoming actions. As a case of study, we consider the world of the Harry Potter fantasy novels and inferring what spell will be cast next given a fragment of a story. Spells act as keywords that abstract actions (e.g. ‘Alohomora’ to open a door) and denote a response to the environment. This idea is used to automatically build HPAC, a corpus containing 82 836 samples and 85 actions. We then evaluate different baselines. Among the tested models, an LSTM-based approach obtains the best performance for frequent actions and large scene descriptions, but approaches such as logistic regression behave well on infrequent actions.

1 Introduction

Natural language processing (NLP) has achieved significant advances in reading comprehension tasks (Chen et al., 2016; Salant and Berant, 2017). These are partially due to embedding methods (Mikolov et al., 2013; Devlin et al., 2018) and neural networks (Rosenblatt, 1958; Hochreiter and Schmidhuber, 1997; Vaswani et al., 2017), but also to the availability of new resources and challenges. For instance, in cloze-form tasks (Hermann et al., 2015; Bajgar et al., 2016), the goal is to predict the missing word given a short context. Weston et al. (2015) presented baBI, a set of proxy tasks for reading comprehension. In the SQuAD corpus (Rajpurkar et al., 2016), the aim is to answer questions given a Wikipedia passage. Kocisky et al. (2018) introduce NarrativeQA, where answering the questions requires to process entire stories. In a related line, Frermann et al. (2017) use fictional crime scene investigation data, from the CSI series, to define a task where the models try to answer the question: ‘who committed the crime?’.

In an alternative line of work, script induction (Schank and Abelson, 1977) has been also a useful approach to evaluate inference and semantic capabilities of NLP systems. Here, a model processes a document to infer new sequences that reflect events that are statistically probable (e.g. go to a restaurant, be seated, check the menu, ...). For example, Chambers and Jurafsky (2008) introduce narrative event chains, a representation of structured knowledge of a set of events occurring around a protagonist. They then propose a method to learn statistical scripts, and also introduce two different evaluation strategies. With a related aim, Pichotta and Mooney (2014) propose a multi-event representation of statistical scripts to be able to consider multiple entities. These same authors (Pichotta and Mooney, 2016) have also studied the abilities of recurrent neural networks for learning scripts, generating upcoming events given a raw sequence of tokens, using BLEU (Papineni et al., 2002) for evaluation.

This paper explores instead a new task: action prediction from natural language descriptions of scenes. The challenge is addressed as follows: given a natural language input sequence describing the scene, such as a piece of a story coming from a transcript, the goal is to infer which action is most likely to happen next.

Contribution We introduce a fictional-domain English corpus set in the world of Harry Potter novels. The domain is motivated by the existence of a variety of spells in these literary books, associated with keywords that can be seen as unambiguous markers for actions that potentially relate to the previous context. This is used to automatically create a natural language corpus coming from hundreds of users, with different styles, interests and writing skills. We then train a number of standard baselines to predict upcoming actions, a task that

requires to be aware of the context. In particular, we test a number of generic models, from a simple logistic regression to neural models. Experiments shed some light about their strengths and weaknesses and how these are related to the frequency of each action, the existence of other semantically related actions and the length of the input story.

2 HPAC: The Harry Potter’s Action prediction Corpus

To build an action prediction corpus, we need to: (1) consider the set of actions, and (2) collect data where these occur. Data should come from different users, to approximate a real natural language task. Also, it needs to be annotated, determining that a piece of text ends up triggering an action. These tasks are however time consuming, as they require annotators to read vast amounts of large texts. In this context, machine comprehension resources usually establish a compromise between their complexity and the costs of building them (Hermann et al., 2015; Kocisky et al., 2018).

2.1 Domain motivation

We rely on an intuitive idea that uses transcripts from the Harry Potter world to build up a corpus for textual action prediction. The domain has a set of desirable properties to evaluate reading comprehension systems, which we now review.

Harry Potter novels define a variety of spells. These are keywords cast by witches and wizards to achieve purposes, such as turning on a light (‘Lumos’), unlocking a door (‘Alohomora’) or killing (‘Avada Kedavra’). They abstract complex and non-ambiguous actions. Their use also makes it possible to build an automatic and self-annotated corpus for action prediction. The moment a spell occurs in a text represents a response to the environment, and hence, it can be used to label the preceding text fragment as a scene description that ends up triggering that action. Table 1 illustrates it with some examples from the original books.

This makes it possible to consider texts from the magic world of Harry Potter as the domain for the action prediction corpus, and the spells as the set of eligible actions.¹ Determining the length of the preceding context, namely *snippet*, that has to be

¹Note that the corpus is built in an automatic way and some occurrences might not correspond to actions, but for example, to a description of the spell or even some false positive samples. Related to this, we have not censored the content of the stories, so some of them might contain adult content.

considered as the scene description is however not trivial. This paper considers experiments (§4) using snippets with the 32, 64, 96 and 128 previous tokens to an action. We provide the needed scripts to rebuild the corpus using arbitrary lengths.²

2.2 Data crawling

The number of occurrences of spells in the original Harry Potter books is small (432 occurrences), which makes it difficult to train and test a machine learning model. However, the amount of available fan fiction for this saga allows to create a large corpus. For HPAC, we used fan fiction (and *only* fan fiction texts) from <https://www.fanfiction.net/book/Harry-Potter/> and a version of the crawler by Milli and Bamman (2016).³ We collected Harry Potter stories written in English and marked with the status ‘completed’. From these we extracted a total of 82 836 spell occurrences, that we used to obtain the scene descriptions. Table 2 details the statistics of the corpus (see also Appendix A). Note that similar to Twitter corpora, fan fiction stories can be deleted over time by users or admins, causing losses in the dataset.⁴

Preprocessing We tokenized the samples with (Manning et al., 2014) and merged the occurrences of multi-word spells into a single token.

3 Models

This work addresses the task as a classification problem, and in particular as a sequence to label classification problem. For this reason, we rely on standard models used for this type of task: multinomial logistic regression, a multi-layered perceptron, convolutional neural networks and long short-term memory networks. We outline the essentials of each of these models, but will treat them as black boxes. In a related line, Kaushik and Lipton (2018) discuss the need of providing rigorous baselines that help better understand the improvement coming from future and complex models, and also the need of not demanding architectural novelty when introducing new datasets.

Although not done in this work, an alternative (but also natural) way to address the task is as a

²<https://github.com/aghie/hpac>

³Due to the website’s Terms of Service, the corpus cannot be directly released.

⁴They also can be modified, making it unfeasible to retrieve some of the samples.

Text fragment	Action
Ducking under Peeves, they ran for their lives, right to the end of the corridor where they slammed into a door - and it was locked. ‘This is it!’ Ron moaned, as they pushed helplessly at the door, ‘We’re done for! This is the end!’ They could hear footsteps, Filch running as fast as he could toward Peeves’s shouts. ‘Oh, move over’, Hermione snarled. She grabbed Harry’s wand, tapped the lock, and whispered, ‘ Alohomora ’.	Unlock the door
And then, without warning, Harry’s scar exploded with pain. It was agony such as he had never felt in all his life; his wand slipped from his fingers as he put his hands over his face; his knees buckled; he was on the ground and he could see nothing at all; his head was about to split open. From far away, above his head, he heard a high, cold voice say, ‘Kill the spare.’ A swishing noise and a second voice, which screeched the words to the night: ‘ Avada Kedavra ’	Kill a target
Harry felt himself being pushed hither and thither by people whose faces he could not see. Then he heard Ron yell with pain. ‘What happened?’ said Hermione anxiously, stopping so abruptly that Harry walked into her. ‘Ron, where are you? Oh, this is stupid’ - ‘ Lumos ’	Turn on a light

Table 1: Examples from the Harry Potter books showing how spells map to reactions to the environment.

Statistics	Training	Dev	Test
#Actions	85	83	84
#Samples	66 274	8 279	8 283
#Tokens (s=32)	2 111 180	263 573	263 937
#Unique tokens (s=32)	33 067	13 075	13 207
#Tokens (s=128)	8 329 531	1 040 705	1 041 027
#Unique tokens (s=128)	60 379	25 146	25 285

Table 2: Corpus statistics: s is the length of the snippet.

special case of language modelling, where the output vocabulary is restricted to the size of the ‘action’ vocabulary. Also, note that the performance for this task is not expected to achieve a perfect accuracy, as there may be situations where more than one action is reasonable, and also because writers tell a story playing with elements such as surprise or uncertainty.

The source code for the models can be found in the GitHub repository mentioned above.

Notation $w_{1:n}$ denotes a sequence of words w_1, \dots, w_n that represents the scene, with $w_i \in V$. $F_\theta(\cdot)$ is a function parametrized by θ . The task is cast as $F : V^n \rightarrow A$, where A is the set of actions.

3.1 Machine learning models

The input sentence $w_{1:n}$ is encoded as a one-hot vector, \mathbf{v} (total occurrence weighting scheme).

Multinomial Logistic Regression Let $\text{MLR}_\theta(\mathbf{v})$ be an abstraction of a multinomial logistic regression parametrized by θ , the output for an input \mathbf{v} is computed as the $\arg \max_{a \in A} P(y = a | \mathbf{v})$, where $P(y = a | \mathbf{v})$ is a *softmax* function, i.e., $P(y = a | \mathbf{v}) = \frac{e^{W_a \cdot \mathbf{v}}}{\sum_{a'} e^{W_{a'} \cdot \mathbf{v}}}$.

MultiLayer Perceptron We use one hidden layer with a rectifier activation function ($\text{relu}(x) = \max(0, x)$). The output is computed as $\text{MLP}_\theta(\mathbf{v}) = \text{softmax}(W_2 \cdot \text{relu}(W \cdot \mathbf{v} + \mathbf{b}) + \mathbf{b}_2)$.

3.2 Sequential models

The input sequence is represented as a sequence of word embeddings, $\mathbf{w}_{1:n}$, where \mathbf{w}_i is a concatenation of an internal embedding learned during the training process for the word w_i , and a pre-trained embedding extracted from GloVe (Pennington et al., 2014)⁵, that is further fine-tuned.

Long short-term memory network (Hochreiter and Schmidhuber, 1997): The output for an element \mathbf{w}_i also depends on the output of \mathbf{w}_{i-1} . The $\text{LSTM}_\theta(\mathbf{w}_{1:n})$ ⁶ takes as input a sequence of word embeddings and produces a sequence of hidden outputs, $\mathbf{h}_{1:n}$ (\mathbf{h}_i size set to 128). The last output of the LSTM_θ , \mathbf{h}_n , is fed to a MLP_θ .

Convolutional Neural Network (LeCun et al., 1995; Kim, 2014). It captures local properties over continuous slices of text by applying a convolution layer made of different filters. We use a wide convolution, with a window slice size of length 3 and 250 different filters. The convolutional layer uses a *relu* as the activation function. The output is fed to a max pooling layer, whose output vector is passed again as input to a MLP_θ .

4 Experiments

Setup All MLP_θ ’s have 128 input neurons and 1 hidden layer. We trained up to 15 epochs using mini-batches (size=16), Adam (lr=0.001) (Kingma and Ba, 2015) and early stopping.

Table 3 shows the macro and weighted F-scores for the models considering different snippet sizes.⁷

⁵<http://nlp.stanford.edu/data/glove.6B.zip>

⁶ n is set to be equal to the length of the snippet.

⁷As we have addressed the task as a classification problem, we will use precision, recall and F-score as the evaluation metrics.

To diminish the impact of random seeds and local minima in neural networks, results are averaged across 5 runs.⁸ ‘Base’ is a majority-class model that maps everything to ‘Avada Kedavra’, the most common action in the training set. This helps test whether the models predict above chance performance. When using short snippets (size=32), disparate models such as our MLR, MLP and LSTMs achieve a similar performance. As the snippet size is increased, the LSTM-based approach shows a clear improvement on the weighted scores⁹, something that happens only marginally for the rest. However, from Table 3 it is hard to find out what the approaches are actually learning to predict.

Snippet	Model	Macro			Weighted		
		P	R	F	P	R	F
-	Base	0.1	1.2	0.2	1.3	11.5	2.4
32	MLR	18.7	11.6	13.1	28.9	31.4	28.3
	MLP	19.1	9.8	10.3	31.7	32.1	28.0
	LSTM	13.7	9.7	9.5	29.1	32.2	28.6
	CNN	9.9	7.8	7.3	24.6	29.2	24.7
64	MLR	20.6	12.3	13.9	29.9	32.1	29.0
	MLP	17.9	9.5	9.8	31.2	32.7	27.9
	LSTM	13.3	10.3	10.2	30.3	33.9	30.4
	CNN	9.8	7.8	7.4	25.0	29.9	25.4
96	MLR	20.4	13.3	14.6	30.3	32.0	29.3
	MLP	16.9	9.5	9.8	30.2	32.6	27.8
	LSTM	14.0	10.5	10.3	30.6	34.5	30.7
	CNN	10.2	7.1	6.9	25.2	29.4	24.4
128	MLR	19.6	12.1	12.9	30.0	31.7	28.2
	MLP	18.9	9.9	10.3	31.4	32.9	28.0
	LSTM	14.4	10.5	10.5	31.3	35.1	31.1
	CNN	8.8	7.8	7.1	24.8	30.2	25.0

Table 3: Macro and weighted F-scores over 5 runs.

To shed some light, Table 4 shows their performance according to a ranking metric, recall at k . The results show that the LSTM-based approach is the top performing model, but the MLP obtains just slightly worse results. Recall at 1 is in both cases low, which suggests that the task is indeed complex and that using just LSTMs is not enough. It is also possible to observe that even if the models have difficulties to correctly predict the action as a first option, they develop certain sense of the scene and consider the right one among their top choices. Table 5 delves into this by splitting the performance of the model into infrequent and frequent actions (above the average, i.e. those that occur more than 98 times in the training set, a total of 20 actions). There is a clear gap between

⁸Some macro F-scores do not lie within the Precision and Recall due to this issue.

⁹For each label, we compute their average, weighted by the number of true instances for each label. The F-score might be not between precision and recall.

the performance on these two groups of actions, with a ~ 50 points difference in recall at 5. Also, a simple logistic regression performs similar to the LSTM on the infrequent actions.

Snippet	Model	R@1	R@2	R@5	R@10
-	Base	11.5	-	-	-
32	MLR	31.4	43.7	60.3	73.5
	MLP	32.1	44.3	61.5	74.9
	LSTM	32.2	44.3	61.5	74.7
	CNN	29.2	41.1	58.1	71.6
64	MLR	32.1	44.9	61.9	74.3
	MLP	32.7	46.0	63.5	76.6
	LSTM	33.9	46.1	63.1	75.7
	CNN	29.9	41.8	59.0	72.2
96	MLR	32.0	44.5	60.7	74.6
	MLP	32.6	45.6	63.4	76.6
	LSTM	34.5	46.9	63.7	76.1
	CNN	29.3	41.9	59.5	72.8
128	MLR	31.7	44.5	61.0	74.3
	MLP	32.9	45.8	63.2	76.9
	LSTM	35.1	47.4	64.4	76.9
	CNN	30.2	42.3	59.6	72.8

Table 4: Averaged recall at k over 5 runs.

Snippet	Model	Frequent		Infrequent			
		F_{we}	R@1	R@5	F_{we}	R@1	R@5
-	Base	3.7	14.5	-	0.0	0.0	-
32	MLR	35.8	37.1	70.5	14.8	9.5	23.0
	MLP	35.9	38.1	71.9	13.2	9.4	21.8
	LSTM	37.1	38.4	71.6	11.7	8.6	23.0
	CNN	33.1	35.5	69.3	7.1	5.2	15.2
64	MLR	36.7	37.9	71.8	14.9	9.9	24.0
	MLP	36.4	39.2	74.5	11.0	7.9	21.6
	LSTM	39.2	40.3	73.0	12.4	9.4	25.4
	CNN	33.9	36.4	70.6	6.9	5.2	15.1
96	MLR	36.4	37.4	70.1	17.1	11.7	25.1
	MLP	36.2	39.1	74.0	11.0	7.9	23.1
	LSTM	39.6	41.1	73.7	12.4	9.6	25.8
	CNN	32.7	35.8	71.6	6.3	4.8	13.7
128	MLR	35.4	37.2	70.5	15.4	10.7	25.0
	MLP	36.5	39.5	74.0	11.1	8.2	22.3
	LSTM	40.3	41.9	74.4	12.3	9.5	26.2
	CNN	33.7	36.9	71.4	6.5	5.0	14.6

Table 5: Performance on *frequent* (those that occur above the average) and *infrequent* actions.

Error analysis¹⁰ Some of the misclassifications made by the LSTM approach were semantically related actions and counter-actions. For example, ‘Colloportus’ (to close a door) was never predicted. The most common mis-classification (14 out of 41) was ‘Alohomora’ (to unlock a door), which was 5 times more frequent in the training corpus. Similarly, ‘Nox’ (to extinguish the light from a wand) was correctly predicted 6 times, meanwhile 36 mis-classifications corre-

¹⁰Made over one of the runs from the LSTM-based approach and setting the snippet size to 128 tokens.

spond to ‘Lumos’ (to light a place using a wand), which was 6 times more frequent in the training set. Other less frequent spells that denote vision and guidance actions, such as ‘Point me’ (the wand acts as a compass pointing North) and ‘Homenum revelio’ (to reveal a human presence) were also mainly misclassified as ‘Lumos’. This is an indicator that the LSTM approach has difficulties to disambiguate among semantically related actions, especially if their occurrence was unbalanced in the training set. This issue is in line with the tendency observed for recall at k . Spells intended for much more specific purposes, according to the books, obtained a performance significantly higher than the average, e.g. F-score(‘Riddikulus’)=63.54, F-score(‘Expecto Patronum’)=55.49 and F-score(‘Obliviate’)=47.45. As said before, the model is significantly biased towards frequent actions. For 79 out of 84 gold actions in the test set, we found that the samples tagged with such actions were mainly classified into one of the top 20 most frequent actions.

Human comparison We collected human annotations from 208 scenes involving frequent actions. The accuracy/F-macro/F-weighted was 39.20/30.00/40.90. The LSTM approach obtained 41.26/25.37/39.86. Overall, the LSTM approach obtained a similar performance, but the lower macro F-score by the LSTM could be an indicator that humans can distinguish within a wider spectrum of actions. As a side note, super-human performance it is not strange in other NLP tasks, such as sentiment analysis (Pang et al., 2002).

5 Conclusion

We explored action prediction from written stories. We first introduced a corpus set in the world of Harry Potter’s literature. Spells in these novels act as keywords that abstract actions. This idea was used to label a collection of fan fiction. We then evaluated standard NLP approaches, from logistic regression to sequential models such as LSTMs. The latter performed better in general, although vanilla models achieved a higher performance for actions that occurred a few times in the training set. An analysis over the output of the LSTM approach also revealed difficulties to discriminate among semantically related actions.

The challenge here proposed corresponded to a fictional domain. A future line of work we are interested in is to test whether the knowledge learned

with this dataset could be transferred to real-world actions (i.e. real-domain setups), or if such transfer is not possible and a model needs to be trained from scratch.

Acknowledgments

This work has received support from the TELEPARES-UDC project (FFI2014-51978-C2-2-R) and the ANSWER-ASAP project (TIN2017-85160-C2-1-R) from MINECO, and from Xunta de Galicia (ED431B 2017/01), and from the European Research Council (ERC), under the European Union’s Horizon 2020 research and innovation programme (FASTPARSE, grant agreement No 714150).

References

- Ondrej Bajgar, Rudolf Kadlec, and Jan Kleindienst. 2016. [Embracing data abundance: Booktest dataset for reading comprehension](#). *arXiv preprint arXiv:1610.00956*.
- Nathanael Chambers and Dan Jurafsky. 2008. [Unsupervised learning of narrative event chains](#). pages 789–797.
- Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. [A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. [Bert: Pre-training of deep bidirectional transformers for language understanding](#). *arXiv preprint arXiv:1810.04805*.
- L. Frermann, S. B. Cohen, and M. Lapata. 2017. [Whodunnit? Crime Drama as a Case for Natural Language Understanding](#). *Transactions of the Association for Computational Linguistics*, 6:1–15.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. [Teaching machines to read and comprehend](#). In *Advances in Neural Information Processing Systems*, pages 1693–1701.
- Sepp Hochreiter and Jürgen Schmidhuber. 1997. [Long short-term memory](#). *Neural computation*, 9(8):1735–1780.
- Divyansh Kaushik and Zachary C. Lipton. 2018. [How much reading does reading comprehension require? a critical investigation of popular benchmarks](#). pages 5010–5015.

- Yoon Kim. 2014. [Convolutional neural networks for sentence classification](#). In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751. Association for Computational Linguistics.
- Diederik Kingma and Jimmy Ba. 2015. [Adam: A method for stochastic optimization](#). In *3rd International Conference for Learning Representation*.
- Tomas Kocisky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gabor Melis, and Edward Grefenstette. 2018. [The narrativeqa reading comprehension challenge](#). *Transactions of the Association for Computational Linguistics*, 6:317–328.
- Yann LeCun, Yoshua Bengio, et al. 1995. [Convolutional networks for images, speech, and time series](#). *The handbook of brain theory and neural networks*, 3361(10):1995.
- Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. [The Stanford CoreNLP natural language processing toolkit](#). In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. [Distributed representations of words and phrases and their compositionality](#). In *Advances in neural information processing systems*, pages 3111–3119.
- Smitha Milli and David Bamman. 2016. [Beyond canonical texts: A computational analysis of fan-fiction](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2048–2053. Association for Computational Linguistics.
- Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. [Thumbs up? sentiment classification using machine learning techniques](#). In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. [Glove: Global vectors for word representation](#). In *Proceedings of the 2014 conference on empirical methods in natural language processing*, pages 1532–1543.
- Karl Pichotta and Raymond Mooney. 2014. [Statistical script learning with multi-argument events](#). In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 220–229. Association for Computational Linguistics.
- Karl Pichotta and Raymond J. Mooney. 2016. [Using sentence-level lstm language models for script inference](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 279–289. Association for Computational Linguistics.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. [Squad: 100,000+ questions for machine comprehension of text](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. Association for Computational Linguistics.
- Frank Rosenblatt. 1958. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.
- S. Salant and J. Berant. 2017. [Contextualized Word Representations for Reading Comprehension](#). *ArXiv e-prints*.
- Roger C Schank and Robert P Abelson. 1977. Scripts: Plans, goals and understanding. *Lawrence Erlbaum*.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. [Attention is all you need](#). In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.
- Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. [Towards ai-complete question answering: A set of prerequisite toy tasks](#). *CoRR*, abs/1502.05698.

A Corpus distribution

Table 6 summarizes the label distribution across the training, development and test sets of the HPAC corpus.

Action	#Training	#Dev	#Test	Action	#Training	#Dev	#Test
AVADA KEDAVRA	7937	986	954	CRUCIO	7852	931	980
ACCIO	4556	595	562	LUMOS	4159	505	531
STUPEFY	3636	471	457	OBLIVIATE	3200	388	397
EXPELLIARMUS	2998	377	376	LEGILIMENS	1938	237	247
EXPECTO PATRONUM	1796	212	242	PROTEGO	1640	196	229
SECTUMSEMPRA	1596	200	189	ALOHOMORA	1365	172	174
INCENDIO	1346	163	186	SCOURGIFY	1317	152	166
REDUCTO	1313	171	163	IMPERIO	1278	159	144
WINGARDIUM LEVIOSA	1265	158	154	PETRIFICUS TOTALUS	1253	175	134
SILENCIO	1145	153	136	REPARO	1124	159	137
MUFFLIATO	1005	108	92	AGUAMENTI	796	84	86
FINITE INCANTATEM	693	90	75	INCARCEROS	686	99	87
NOX	673	82	80	RIDDIKULUS	655	81	88
DIFFINDO	565	90	82	IMPEDIMENTA	552	88	79
LEVICORPUS	535	63	68	EVANESCO	484	53	59
SONORUS	454	66	73	POINT ME	422	57	69
EPISKEY	410	55	59	CONFRINGO	359	52	48
ENGORGIO	342	52	41	COLLOPORTUS	269	26	41
RENNERVATE	253	24	33	PORTUS	238	22	31
TERGEO	235	23	26	MORSMORDRE	219	29	38
EXPULSO	196	23	20	HOMENUM REVELIO	188	30	24
MOBILICORPUS	176	20	14	RELASHIO	174	20	27
LOCOMOTOR	172	24	19	AVIS	166	17	29
RICTUSEMPRA	159	16	26	IMPERVIUS	149	26	13
OPPUGNO	144	18	7	FURNUNCULUS	137	20	20
SERPENSORTIA	133	14	15	CONFUNDO	130	17	21
LOCOMOTOR MORTIS	127	14	15	TARANTALLEGRA	126	11	17
REDUCIO	117	13	22	QUIETUS	108	15	17
LANGLOCK	99	12	19	GEMINIO	78	5	10
FERULA	78	6	10	ORCHIDEOUS	76	7	5
DENSAUGEO	67	13	8	LIBERACORPUS	63	7	5
APARECIUM	63	14	10	ANAPNEO	62	6	5
FLAGRATE	59	4	11	DELETRIOUS	59	12	6
OBSCURO	57	11	7	PRIOR INCANTATO	56	4	3
DEPRIMO	51	2	2	SPECIALIS REVELIO	50	11	6
WADDIWASI	45	5	8	PROTEGO TOTALUM	44	9	5
DURO	36	4	4	SALVIO HEXIA	36	8	5
DEFODIO	34	2	6	PIERTOTUM LOCOMOTOR	30	4	3
GLISSEO	26	4	3	MOBILIARBUS	25	3	4
REPELLO MUGGLETUM	23	2	5	ERECTO	23	7	5
CAVE INIMICUM	19	5	2	DESCENDO	19	0	1
PROTEGO HORRIBILIS	18	7	5	METEOLOJINX RECANTO	10	3	1
PESKIPIKSI PESTERNOMI	7	0	0				

Table 6: Label distribution for the HPAC corpus