Appendices for Baleen

A Data Details

Table 6: Sizes of the splits of the datasets used in this work.

Multi-Hop Dataset	**Train**	**Dev**	**Test**
HotPotQA	90,447	7,405	7,405
HoVer	18,171	4,000	4,000

As our retrieval corpus for both HotPotQA and HoVer, we use the Wikipedia dump released by Yang et al. [29] from Oct 2017.[4] This is the Wikipedia dump used officially for HotPotQA and for HoVer. For each Wikipedia page, this corpus contains only the first paragraph and the passages are already divided into individual sentences. It contains approximately 5M passages (1.5 GiB uncompressed). We use the official data splits for both datasets, described in Table 6.

B Baleen implementation & hyperparameters

We implement Baleen using Python 3.7 and PyTorch 1.6 and rely extensively on the HuggingFace Transformers library [26].[5] We train and test with automatic mixed precision that is built into PyTorch.

B.1 FLIPR retriever

Our implementation of FLIPR is an extension of the ColBERT [14] open-source code,[6] where we primarily modify the retrieval modeling components (e.g., adding focused late interaction and including condensed fact tokens in the query encoder).

For FLIPR, we fine-tune a BERT-base model (110M parameters). For each round of training, we initialize the model parameters from a ColBERT model previously trained on the MS MARCO Passage Ranking task [18]. To train the single-hop retriever used to initiate the supervision procedure of §3.2, we follow the training strategy of Khattab et al. [15]. In particular, we use this *out-of-domain* ColBERT model to create training triples, and then we train our retriever (in this case, FLIPR for first-hop) with these triples. Once we have this first-hop model, the rest of the procedure follows Algorithm 1 for latent hop ordering.

Table 7: Hyperparameters for Baleen's FLIPR on HoVer and HotPotQA.

Hyperparameter	**HoVer**	**HotPotQA**
Learning rate	3×10^{-6}	3×10^{-6}
Embedding Dimension	128	128
Batch size (triples)	48	48
Maximum Passage Length	256	256
Maximum Query Length: query/overall	64/512	64/512
Training steps (round #1; per hop)	10k, 5k, 5k, 5k	20k, 20k
Training steps (round #2)	10k	40k
Negative Sampling Depth (for each hop)	1000	1000
Context Sampling Depth (from each hop)	5	5
Positive Sampling Depth (round #1; per hop)	20, all, all, all	20, all
Positive Sampling Depth (round #2; per hop)	10, 10, 10, all	10, all
FAISS centroids (probed)	8192 (16)	8192 (16)
FAISS results per vector: training/inference	256/512	256/512
Top-k Passages Per Hop	25, 25, 25, 25	10, 40

Table 7 describes our hyperparameters for FLIPR. We manually explored a limited space of hyper-parameters in preliminary experiments, tuning Retrieval@k Accuracy, with k=100 for HoVer and

[4] `https://hotpotqa.github.io/wiki-readme.html`
[5] `https://github.com/huggingface/transformers`
[6] `https://github.com/stanford-futuredata/ColBERT`

k=20 for HotPotQA, while also being cognizant of downstream Psg-EM and Sent-EM. We expect that a larger tuning budget would lead to further gains, which is consistent with the fact that our Psg-EM and Sent-EM on the *held-out* leaderboard test set are 1.0 and 0.6 points *higher* than the public validation set with which we developed our methods. We adopt the default learning rate from ColBERT, namely 3×10^{-6}. We set the embedding dimension to the default $d = 128$ and use a batch size of 48 triples. We truncate passages to 256 tokens. For the query encoder, we truncate queries to 64 tokens and allow up to 512 tokens in total, particularly for the condensed facts or reranker context passages from the previous hops. For FAISS [12] end-to-end retrieval (see Khattab and Zaharia [14] for use with late interaction models),[7] we use the query component and then apply (focused) late interaction on both the query and facts embeddings.

B.2 Condensers and rerankers

For the two-stage condenser, we train two ELECTRA-large models, one per stage. We simply use a `[MASK]` token to separate the facts/sentences, although we expect that any other special-token choice would work similarly provided enough training data is available. We train the first-stage condenser with $\langle query, positivepassage, negativepassage \rangle$ triple. We use a cross-entropy loss over the individual sentences of both passages, where the model has to select the positive sentence out of $\langle positivesentence, *negatives \rangle$ for each positive. We average the cross-entropy loss per example, then across examples per batch. We train the second-hop condenser over a set of 7–9 facts, some positive and others (sampled) negative, using a linear combination of cross-entropy loss for each positive fact (against all negatives) and binary cross-entropy loss for each individual fact.

Table 8: Hyperparameters for Baleen's condensers on HoVer and HotPotQA.

Hyperparameter	**HoVer**	**HotPotQA**
Learning rate	1×10^{-5}	1×10^{-5}
Batch size	64	64
Maximum Sequence Length	512	512
Warmup Steps	1000	1000
Training steps (stage #1)	5k	10k
Training steps (stage #2)	5k	10k
Negative Sampling Depth (stage #1; per hop)	20, 20, 20, 20	10, 30
Negative Sampling Depth (stage #2; for each hop)	10	10
Context Sampling Depth (from each hop)	5	5
Positive Sampling Depth (stage #1; per hop)	10, 10, 10, all	10, all
Positive Sampling Depth (stage #2; for each hop)	10	10
Facts fed to stage # 2: training (inference)	7–9 (9)	7–9 (9)

Table 8 describes our hyperparameters for the condensers.

Claim Verification For HoVer, we train an ELECTRA-large model for claim verification. The input contains the query and the condensed facts and the output is binary (supported/unsupported). We use batches of 16 examples and train for 20,000 steps, but otherwise adopt similar hyperparameters to the condensers.

Reranker We similarly use ELECTRA-large for the rerankers. The input contains the query and one reranker-selected passage for each of the previous hops as well as one passage to consider for the current hop. We adopt the same positive, negative, and context sampling as the first-stage condenser, as well as other hyperparameters, but we allow twice the training budget since there is only one stage for reranking.

Hybrid condenser/reranker implementation We train a single retriever on top of the first-round retrievers for the condenser and reranker Baleen architectures. During inference, we run two independent pipelines, one with the condenser and the other with the reranker. We merge the overall top-100 results by taking the top-13 and top-12 per hop from both retrievers without duplicates, for a total of 25x4=100 unique passages.

[7] `https://github.com/facebookresearch/faiss/`

B.3 Resources used

We conducted our experiments primarily using internal cluster resources. We use four 12GB Titan V GPUs for retrievers and four 32GB V100 GPUs for condensers, rerankers, and readers. Training FLIPR on the four-hop HoVer dataset requires five (4+1) short training runs, for a *total* time of approximately five hours. Similarly, we encode and index the corpus five times in total (four intermediate and one final time) less than six hours in total. Retrieving positives and negatives for training from the index four times consumes a total of less than three hours. All four hops of retrieval on the validation set with the final FLIPR model take a total of a little over one hour. Training both condenser stages for 5k steps each and training the claim verification reader for 20k steps takes a total of less than eight hours. We use python scripts for pre- and post-processing (e.g., for LHO) and run the condensers during evaluation, which generally consume only minutes each.

Our FLIPR retriever adopts a fine-grained late interaction paradigm like ColBERT (see §2), so our memory footprint is relatively large, as it involves storing a small 256-byte (2-byte 128 dimensions) vector per token. The uncompressed index is about 83 GiBs. We note that the authors of ColBERT Khattab and Zaharia [14] have recently released a quantized implementation that can reduce the storage per vector 4–8 fold and reducing the storage space of dense retrieval methods through compression and quantization while preserving accuracy is an active area of research [10; 28], with recent encouraging results.

C The effect of condensing on the context lengths

We compare our condenser architecture of Baleen to a reranker after four hops on HoVer. We find that the average context per query is 91 words for Baleen's condenser architecture versus 325 words for the reranking ablation, on average. This 3.6$\times$ improvement for Baleen's condenser suggests that for tasks with even more hops, a condenser approach would be less likely to overwhelm typical maximum sequence lengths of existing Transformer architectures.

Supplementary Material References for the Appendix

[1] Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. Learning to retrieve reasoning paths over wikipedia graph for question answering. *ICLR 2020*, 2020.

[2] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. arXiv preprint arXiv:1704.00051, 2017. URL `https://arxiv.org/abs/1704.00051`.

[3] Jifan Chen and Greg Durrett. Understanding dataset design choices for multi-hop reasoning. *arXiv preprint arXiv:1904.12106*, 2019.

[4] Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. Decontextualization: Making sentences stand-alone. *Transactions of the Association for Computational Linguistics*, 9:447–461, 2021.

[5] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[7] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*, 2018.

[8] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020.

[9] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and A. Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *ArXiv*, abs/2011.01060, 2020.

[10] Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Sebastian Riedel, and Edouard Grave. A memory efficient baseline for open domain question answering. *arXiv preprint arXiv:2012.15156*, 2020.

[11] Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. HoVer: A dataset for many-hop fact extraction and claim verification. *arXiv preprint arXiv:2011.03088*, 2020.

[12] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 2019.

[13] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

[14] Omar Khattab and Matei Zaharia. ColBERT: Efficient and effective passage search via contextualized late interaction over BERT. *SIGIR 2020*, 2020.

[15] Omar Khattab, Christopher Potts, and Matei Zaharia. Relevance-guided supervision for openqa with colbert. *arXiv preprint arXiv:2007.00814*, 2020.

[16] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*, 2019.

[17] Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive NLP tasks. *arXiv preprint arXiv:2005.11401*, 2020.

[18] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human-generated MAchine reading COmprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

[19] Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vassilis Plachouras, Tim Rocktäschel, et al. Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*, 2020.

[20] Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. Answering complex open-domain questions through iterative query generation. In *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. URL `https://nlp.stanford.edu/pubs/qi2019answering.pdf`.

[21] Peng Qi, Haejun Lee, Oghenetegiri Sido, Christopher D Manning, et al. Retrieve, rerank, read, then iterate: Answering open-domain questions of arbitrary complexity from text. *arXiv preprint arXiv:2010.12527*, 2020.

[22] James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. Fever: a large-scale dataset for fact extraction and verification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, 2018.

[23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL `http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf`.

[24] Haoyu Wang, Mo Yu, Xiaoxiao Guo, Rajarshi Das, Wenhan Xiong, and Tian Gao. Do multi-hop readers dream of reasoning chains? *arXiv preprint arXiv:1910.14520*, 2019.

[25] Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302, 2018.

[26] Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, 2020.

[27] Wenhan Xiong, Xiang Lorraine Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Wen-tau Yih, Sebastian Riedel, Douwe Kiela, et al. Answering complex open-domain questions with multi-hop dense retrieval. *arXiv preprint arXiv:2009.12756*, 2020.

[28] Ikuya Yamada, Akari Asai, and Hannaneh Hajishirzi. Efficient passage retrieval with hashing for open-domain question answering, 2021.

[29] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018.

[30] Chen Zhao, Chenyan Xiong, Corby Rosset, Xia Song, Paul Bennett, and Saurabh Tiwary. Transformer-xh: Multi-evidence reasoning with extra hop attention. In *International Conference on Learning Representations*, 2019.