Model Cascades for Efficient Image Search

Robert Hönig*
robhoenig@gmail.com
ETH Zürich
Switzerland

Mingyuan Chi*
minchi@ethz.ch
ETH Zürich
Switzerland

ABSTRACT

Modern neural encoders offer unprecedented text-image retrieval (TIR) accuracy. However, their high computational cost impedes an adoption to large-scale image searches. We propose a novel image ranking algorithm that uses a cascade of increasingly powerful neural encoders to progressively filter images by how well they match a given text. Our algorithm reduces lifetime TIR costs by over 3x.

CCS CONCEPTS

• **Information systems** → *Retrieval models and ranking*; *Top-k retrieval in databases*; **Combination, fusion and federated search**.

KEYWORDS

neural networks, text-image retrieval, cascaded models

ACM Reference Format:
Robert Hönig and Mingyuan Chi. 2023. Model Cascades for Efficient Image Search. In *Proceedings of (Submitted to DEEM '23).* ACM, New York, NY, USA, 4 pages. https://doi.org/XXXXXXX.XXXXXXX

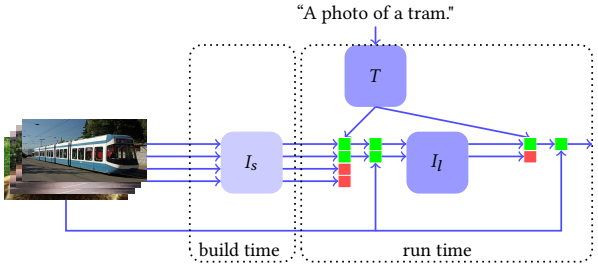

Figure 1: Schematic of our algorithm for a 2-level cascade $[I_s, I_l]$. In this example, encoder I_s computes embeddings $V_{\mathcal{D}}$ of all four images (leftmost four squares) at build time. At runtime, the images that correspond to two the highest-ranking embeddings (green) are processed by encoder I_l that produces embeddings $V_{\mathcal{D}_2}$ of higher quality. Finally, we rerank the top-2 images with $V_{\mathcal{D}_2}$ to output the highest-ranking image.

1 INTRODUCTION

Search engines are the most widely used tool for information retrieval (IR) on the internet — Google alone processes over 8.5 billion searches a day [12]. A search engine takes as input some query q and returns a list of documents $\mathcal{D}$ ranked by how well they match q. Keyword-based search ranks results by naively matching query keywords with documents. Semantic search tries to improve on keyword-based search by matching queries to documents based on their meaning. A fruitful domain for semantic search is TIR, where documents are images and queries are texts. New semantic search engines for TIR leverage recent advances in deep learning for processing images and natural language [7, 11, 15, 18]. Typically, these engines use neural networks to construct an image encoder I and a text encoder T that process text q and each image $d \in \mathcal{D}$ into embeddings $\boldsymbol{v}_q = T(q)$ and $\boldsymbol{V}_{\mathcal{D}} = \{\boldsymbol{v}_d = I(d) : d \in \mathcal{D}\}$ that capture their semantics. Then, the engines rank images in $\mathcal{D}$ by some similarity measure of $\boldsymbol{v}_d$ and $\boldsymbol{v}_q$. In large-scale search scenarios, $\mathcal{D}$ may contain several million documents. This makes it computationally expensive to compute embeddings for all documents.

We seek to lower this computational cost while preserving search quality. To this end, we measure search quality as Recall@k, which denotes the fraction of searches that include the desired result in the top-k results. Even small increases in k can significantly improve the Recall@k [13, 21]. Hence, for $g \gg k$, the top-k results of a large and expensive encoder I_l are likely included in the top-g results of a small and cheap encoder I_s. This observation leads to our main idea: *At build time, pre-compute $\boldsymbol{V}_{\mathcal{D}}$ with I_s. Then, at runtime, to handle a query q, retrieve the top-m results $\mathcal{D}_m \subset \mathcal{D}$ for some $m \gg k$, recompute $\boldsymbol{V}_{\mathcal{D}_m}$ with I_l and return the top-k results.* This idea, illustrated in Figure 1), naturally extends to a cascade of r progressively larger encoders that compute progressively smaller sets $\boldsymbol{V}_1 \ldots, \boldsymbol{V}_r$.

In practice, it is possible for over 90% of all documents in $\mathcal{D}$ to never be included in any search result over the lifetime of a large-scale search engine [16]. This means that our technique would evaluate I_l on less than 10% of $\mathcal{D}$, resulting in significant lifetime computational savings. In this work we make the following contributions:

- We introduce a novel cascading algorithm for fast TIR.
- We show that our algorithm speeds up TIR on standard benchmarks by over 3x at no reduction in search quality.
- We investigate the benefits of deep cascades and demonstrate a 2x reduction in query latency.

*Both authors contributed equally to this research.

Submitted to DEEM '23, June 18, 2023, Seattle, WA, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/XXXXXXX.XXXXXXX

2 RELATED WORK

Model cascading is a recurrent theme in the literature on efficient machine learning (ML) systems. FrugalML [2] minimizes access costs of ML APIs by cascading two calls to a cheap API and to an expensive API. NoScope [8] speeds up object detection in videos by splitting a reference model into a squence of two specialized models. Model cascades have also been applied to facial key point estimation [10], pedestrian detection [1] and other domains.

Recent work on encoders for TIR is dominated by transformer-based bi-encoders (BEs) [4, 13, 21] and cross-encoders (CEs) [9, 19, 22]. BEs process images and texts with separate encoders, whereas CEs also add cross-connections between the encoders. Hence, CEs are more powerful, but need to recompute $\boldsymbol{V}_{\mathcal{D}}$ for new queries. This makes them impractical for large-scale searches and unsuitable for our idea. Therefore, we focus on BEs.

Several methods for fast TIR with CEs have been developed: VLDeformer [23] trains a decomposable CE that can be used as a BE at inference time with minimal loss in quality. CrispSearch [6], LightningDot [17] and "Retrieve Fast, Rerank Smart" [5] all introduce two-level sequences of a BE whose results can be cached for approximate inference and a CE for precise inference on a subset of the BE results. This is similar to our idea but differs in two key ways: First, we consider arbitrarily deep model cascades, whereas these approaches are fundamentally limited to two models. Second, we target BE inference instead of CE inference. In fact, this suggests that our approach could complement these existing techniques as the BE model in their first stage for even faster TIR.

3 MODELS AND METHODS

3.1 Cascaded Search

Let $\mathcal{D}$ be a collection of n images that we want to query with a cascade of BEs. Consider a cascade of image encoders $I = [I_s, I_1, \ldots, I_r]$ that all use the same text encoder T. We propose Algorithm 1 to query $\mathcal{D}$ by ranking all images with I_s and subsequently the top m_j images with I_j. Note that with $r = 0$, Algorithm 1 reduces to a standard BE search.

Computational cost. Assume that function Query in Algorithm 1 invoked q times and denote the computational cost of Algorithm 1 with $C(I, q)$. We want to minimize the lifetime computational cost of Algorithm 1, that is $C(I, q)$ as $q \to \infty$. We can decompose $C(I, q)$ into the sum of the lifetime image encoding cost $a(I, q)$ and some term $b(q)$ that is independent of I and thus irrelevant for optimization over I. Next, we formalize our introductory observation on the set of a search engine's lifetime search results into the following key assumption:

Assumption 1. *For $q \in \mathbb{N}$, let $S_q \subset \mathcal{D}$ be the set of all images pushed to* Top *in query q. Then, $\frac{1}{n}\left|\bigcup_{q\in\mathbb{N}} S_q\right| =: f \ll 1$.*

If $I_s, I_1, \ldots, I_r$ have costs $t_s < t_1 < \ldots < t_r$, then Assumption 1 implies that

$$a(I, q) = nt_s + fn\sum_{i=1}^{r} t_i.$$

Hence, the 2-level cascade $[I_s, I_1]$ is cheaper than the 1-level cascade $[I_1]$ if the speedup factor $(t_s + ft_1)/t_1$ exceeds 1.

Algorithm 1 Cascaded Search. Here, $\text{Rank}(\mathcal{I}, \boldsymbol{V}, \boldsymbol{t})$ sorts the images in $\mathcal{I}$ by the cosine similarity of their encodings $\boldsymbol{V}$ with text encoding $\boldsymbol{t}$.

Input: $[I_s, I_1, \ldots, I_r]$, $m_1 > \ldots > m_r \in \mathbb{N}$, $\mathcal{D}$
Init: for $c \in \mathcal{D}$ **do** $\boldsymbol{V}_s[c] \longleftarrow I_s(c)$
Function Query(text)
  Top $\longleftarrow$ Rank$(\mathcal{D}, \boldsymbol{V}_s, T(\text{text}))$
  for $j = 1$ **to** r **do**
    for $c \in \text{Top}[1 \ldots m_j]$ **do** $\boldsymbol{V}_j[c] \xleftarrow{\text{if empty}} I_j(c)$
    Top $\longleftarrow$ Rank$\big(\text{Top}[1 \ldots m_j], \boldsymbol{V}_j, T(\text{text})\big)$
  end for
  return Top[1]
EndFunction

We note that Assumption 1 implies no computational advantage of the $(r+1)$-level cascade I with $r > 1$ over the equally powerful 2-level cascade $I' = [I_s, I_r]$ with $m'_r = m_1$. However, if q is low enough that $\boldsymbol{V}$ is not hit, then the $(r+1)$-level cascade I speeds up individual queries by a factor of

$$m_1 t_r / \sum_{i=1}^{r} m_i t_i \tag{1}$$

This is useful, because unlike an uncascaded model $[I_r]$ that executes the expensive image encoder I_r only during build time, the 2-level cascade I' has a $m_1 t_r$ runtime overhead when $\boldsymbol{V}$ is not hit. Hence, deep cascades can mitigate the increased latency of early queries in 2-level cascades.

3.2 Creating the Cascade

Table 1: Recall@k in % and lifetime speedup of the 2-level cascade [ViT-B/32, ViT-B/16] over the uncascaded baseline [ViT-B/16].

Dataset	Method	R@1	R@5	R@10	Speedup
MSCOCO	No Cascade	30.1	54.2	64.6	1x
	Cascade	+0.2	+0.4	+0.5	3.2x
Flickr30k	No Cascade	29.9	52.0	61.3	1x
	Cascade	+0.8	+2.0	+2.4	3.2x

We apply our proposed methods to CLIP [13], a powerful transformer-based text-image BE. CLIP uses the GPT-2 architecture [14] for the text encoder, the vision transformer (ViT) [3] architecture for the image encoder and matches images to texts by the cosine similarity of their embeddings. We create a cascade $[I_s, I_1, \ldots, I_r]$ from publicly available trained CLIP image encoders of different sizes.

4 EXPERIMENTS

4.1 Experimental Setup

Metrics Given a dataset $\mathcal{D}$ of image-caption pairs we measure the Recall@k (R@k) metric as the fraction of captions in $\mathcal{D}$ whose corresponding image is among the top-k search results. In line with the IR literature, we report the Recall@k for

$k \in \{1, 5, 10\}$. In addition, we report for 2-level cascades the lifetime speedup and for deeper cascades the query speedup as discussed in Section 3.1. We run all experiments on an Intel i7-11800H CPU at 2.30 GHz with turboboost disabled and compute speedups by measuring the total CPU time of queries.

Datasets We evaluate our algorithm on the MSCOCO validation dataset with 5k samples and on the Flickr30k dataset with 32k samples.

Parameters We set the top-m value of encoder I_1 to $m_1 = 50$ and assume a lifetime return fraction of $f = 0.1$.

4.2 2-level cascades

We use the Huggingface [20] CLIP implementation with a ViT-B/16 image encoder as our uncascaded baseline $[I_1]$. We use the faster ViT-B/32 image encoder as I_s to create the 2-level cascade $[I_s, I_1]$. Table 1 shows empirical results. The cascaded model reduces lifetime computational costs threefold. Surprisingly, the cascaded model achieves at the same time consistently higher Recall@k than the uncascaded model. One explanation may be that ViT-B/32 initially processes input images into 32x32 tiles. Since this tiling is more coarse-grained than the 16x16 tiling used by ViT-B/16, it may offer superior approximate filtering of search results. Hence, I_s could determine the top m_1 images more effectively than I_1. Further research is needed to explain why 2-level cascades show superior Recall@k.

4.3 3-level cascades

Table 2: Recall@k in % and query speedup of the 3-level cascade [ViT-B/32, ViT-B/16, ViT-L/14] with $m_2 = 10$ over the 2-level cascade [ViT-B/32, ViT-L/14].

Dataset	Method	R@1	R@5	R@10	Speedup
MSCOCO	No Cascade	32.5	57.2	68.1	1x
	Cascade	+0.5	+0.2	-3.0	2.0x
Flickr30k	No Cascade	35.3	58.5	67.4	1x
	Cascade	+1.0	+0.0	-3.7	2.0x

As noted in Section 3.1, n-level cascades offer no reduced lifetime costs over 2-level cascades, but may speed up individual queries. This is important for large image encoders that slow down queries, such as the ViT-L/14 encoder that is 3.3x slower than ViT-B/16. Therefore, we introduce the 2-level cascade [ViT-B/32, ViT-L/14] and compare it against the 3-level cascade [ViT-B/32, ViT-B/16, ViT-L/14]. Concretely, we set a target speedup of 2x and use Equation (1) to determine the corresponding number m_2 of top ranked images on which Algorithm 1 should execute ViT-L/14. This yields $m_2 = m_1 \left(\frac{1}{2} - \frac{t_1}{t_2}\right) = 50 \left(\frac{1}{2} - \frac{1}{3.3}\right) \approx 10$. Table 2 reports the empirically measured query speedups and the change in Recall@k of the 3-level cascade. Similarly to Section 4.2, the deeper cascade offers superior predictions. However, for Recall@10 the predictions become significantly worse. This is because Algorithm 1 only uses ViT-L/14 to rerank the top $m_2 = 10$ images, so the set of the top 10 images stays unchanged. Hence, for $m_2 = 10$, the cascade [ViT-B/32, ViT-B/16, ViT-L/14] is equivalent to the less powerful cascade [ViT-B/32, ViT-B/16] with respect to the Recall@10 metric.

5 CONCLUSION

Our experiments show that Algorithm 1 can lower lifetime computational search costs by over 3x at no reduction in search quality. At the same time, we show that deeper model cascades can mitigate the increase in latency of early queries.

However, single-digit speedups may not sufficiently reduce computational costs to economically rank large-scale image databases with expensive transformer-based BEs. Instead, a practitioner may use traditional search engines to retrieve the top-k images and apply a neural search cascade on top of it. This heterogeneous cascade may offer a viable path towards the integration of state-of-the-art neural networks with established image search platforms.

It is important to note that all our observations rely on Assumption 1. While we have provided anecdotal evidence to support our choice of the lifetime return fraction as $f = 10\%$, different search scenarios likely vary in f and achieve accordingly different speedups.

REFERENCES

[1] Zhaowei Cai, Mohammad Saberian, and Nuno Vasconcelos. 2015. Learning complexity-aware cascades for deep pedestrian detection. In *Proceedings of the IEEE international conference on computer vision*. 3361–3369.

[2] Lingjiao Chen, Matei Zaharia, and James Y Zou. 2020. Frugalml: How to use ml prediction apis more accurately and cheaply. *Advances in neural information processing systems* 33 (2020), 10685–10696.

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[4] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. 2017. Vse++: Improving visual-semantic embeddings with hard negatives. *arXiv preprint arXiv:1707.05612* (2017).

[5] Gregor Geigle, Jonas Pfeiffer, Nils Reimers, Ivan Vulić, and Iryna Gurevych. 2021. Retrieve fast, rerank smart: Cooperative and joint approaches for improved cross-modal retrieval. *arXiv preprint arXiv:2103.11920* (2021).

[6] Zhiming Hu, Lan Xiao, Mete Kemertas, Caleb Phillips, Iqbal Mohomed, and Afsaneh Fazly. 2022. CrispSearch: low-latency on-device language-based image retrieval. In *Proceedings of the 13th ACM Multimedia Systems Conference*. 62–72.

[7] Surbhi Jain and Joydip Dhar. 2017. Image based search engine using deep learning. In *2017 Tenth International Conference on Contemporary Computing (IC3)*. IEEE, 1–7.

[8] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529* (2017).

[9] Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. 2020. Unicoder-VL: A Universal Encoder for Vision and Language by Cross-Modal Pre-Training. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 07 (Apr. 2020), 11336–11344. https://doi.org/10.1609/aaai.v34i07.6795

[10] Haoxiang Li, Zhe Lin, Xiaohui Shen, Jonathan Brandt, and Gang Hua. 2015. A convolutional neural network cascade for face detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5325–5334.

[11] Richa Mishra and Surya Prakash Tripathi. 2021. Deep learning based search engine for biomedical images using convolutional neural networks. *Multimedia Tools and Applications* 80, 10 (2021), 15057–15065.

[12] Maryam Mohsin. 2022. 10 GOOGLE SEARCH STATISTICS YOU NEED TO KNOW IN 2022 [INFOGRAPHIC]. https://www.oberlo.com/blog/google-search-statistics. Accessed: 2022-12-02.

[13] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*. PMLR, 8748–8763.

[14] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[15] Lester Solbakken. 2021. Text-to-image search with Vespa. https://blog.vespa.ai/text-image-search/. Accessed: 2022-12-02.

[16] Tim Soulo. 2020. Search traffic study. https://ahrefs.com/blog/search-traffic-study/. Accessed: 2022-12-02.

[17] Siqi Sun, Yen-Chun Chen, Linjie Li, Shuohang Wang, Yuwei Fang, and Jingjing Liu. 2021. Lightningdot: Pre-training visual-semantic embeddings for real-time image-text retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 982–997.

[18] Hima Tammineedi. 2021. Evertrove - We made a usable ML-powered image search using OpenAI's CLIP - search millions of images. https://www.reddit.com/r/MachineLearning/comments/lcjizm/p_evertrove_we_made_a_usable_mlpowered_image/. Accessed: 2022-12-02.

[19] Hao Tan and Mohit Bansal. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 5100–5111. https://doi.org/10.18653/v1/D19-1514

[20] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771* (2019).

[21] Yan Zeng, Xinsong Zhang, and Hang Li. 2021. Multi-Grained Vision Language Pre-Training: Aligning Texts with Visual Concepts. *arXiv preprint arXiv:2111.08276* (2021).

[22] Yan Zeng, Xinsong Zhang, Hang Li, Jiawei Wang, Jipeng Zhang, and Wangchunshu Zhou. 2022. X^2-VLM: All-In-One Pre-trained Model For Vision-Language Tasks. *arXiv preprint arXiv:2211.12402* (2022).

[23] Lisai Zhang, Hongfa Wu, Qingcai Chen, Yimeng Deng, Joanna Siebert, Zhonghua Li, Yunpeng Han, Dejiang Kong, and Zhao Cao. 2022. VLDeformer: Vision–Language Decomposed Transformer for fast cross-modal retrieval. *Knowledge-Based Systems* 252 (2022), 109316.