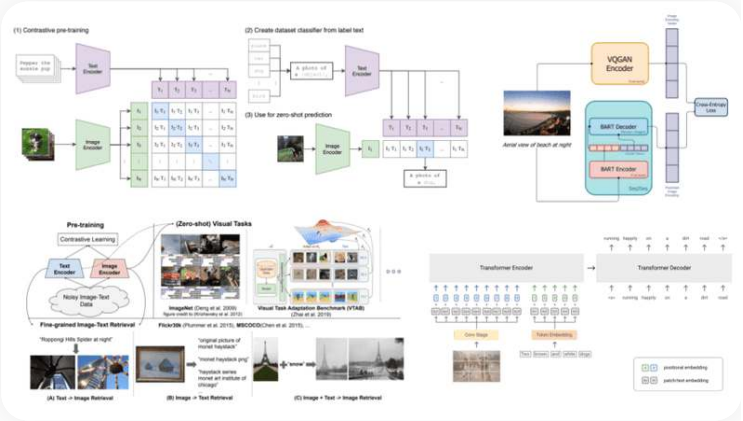
Vision Language models: towards multi-modal deep learning



Sergios Karagiannakos on 2022-03-03 · 12 mins

Natural Language Processing

Attention and Transformers Computer Vision

Multimodal learning refers to the process of **learning representations from different types of modalities using the same model**. Different modalities are characterized by different statistical properties. In the context of machine learning, input modalities include images, text, audio, etc. In this article, we will discuss only images and text as inputs and see how we can build Vision-Language (VL) models.

Vision-language tasks

Vision-language models have gained a lot of popularity in recent years due to the number of potential applications. We can roughly categorize them into 3 different areas. Let's explore them along with their

Generation tasks

- **Visual Question Answering (VQA)** refers to the process of providing an answer to a question given a visual input (image or video).
- **Visual Captioning (VC)** generates descriptions for a given visual input.
- **Visual Commonsense Reasoning (VCR)** infers common-sense information and cognitive understanding given a visual input.
- **Visual Generation (VG)** generates visual output from a textual input, as shown in the image.

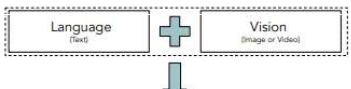
Source: OpenAI's blog

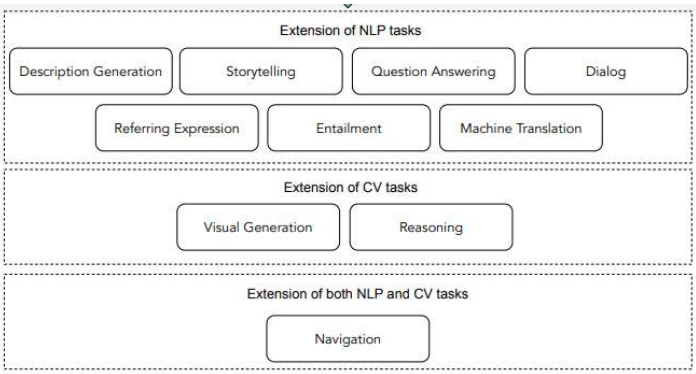
Classification tasks

- **Multimodal Affective Computing (MAC)** interprets visual affective activity from visual and textual input. In a way, it can be seen as multimodal sentiment analysis.
- **Natural Language for Visual Reasoning (NLVR)** determines if a statement regarding a visual input is correct or not.

Retrieval tasks

- **Visual Retrieval (VR)** retrieves images based only on a textual description.
- **Vision-Language Navigation (VLN)** is the task of an agent navigating through a space based on textual instructions.
- **Multimodal Machine Translation (MMT)** involves translating a description from one language to another with additional visual information.





Taxonomy of popular visual language tasks [1]

Depending on the task at hand, different architectures have been proposed over the years. In this article, we will explore some of the most popular ones.

BERT-like architectures

Given the incredible rise of transformers in NLP, it was inevitable that people would also try to apply them in VL tasks. The majority of papers have been used some version of BERT [2], resulting in a simultaneous explosion of BERT-like multimodal models: VisualBERT [3], ViLBERT [4], Pixel-BERT [5], ImageBERT [6], VL-BERT [7], VD-BERT [8], LXMERT [9], UNITER [10].

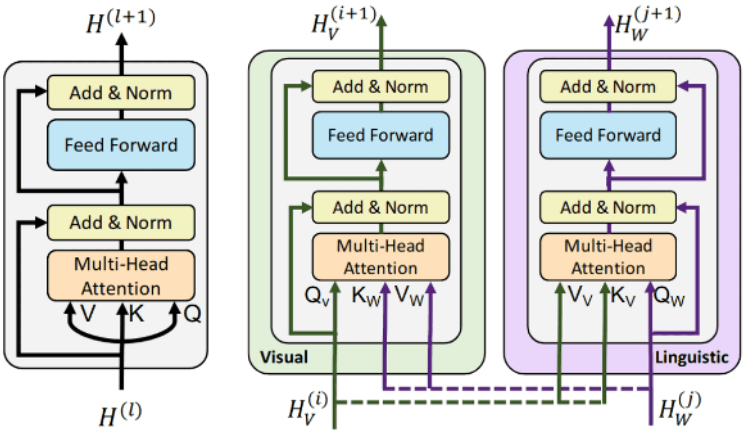
They are all based on the same idea: they process language and images at the same time with a transformer-like architecture. We generally divide them into two categories: two-stream models and single-stream models.

Two-stream models: ViLBERT

Two-stream model is a literature term that refers to VL models which process text and images using two separate modules. ViLBERT [4] and LXMERT [9] fall into this category.

ViLBERT [4] is trained on image-text pairs. The text is encoded with the standard transformer process using tokenization and positional embeddings. It is then processed by the self-attention modules of the transformer. Images are decomposed into non-overlapping patches projected in a vector, as in vision transformer's patch embeddings.

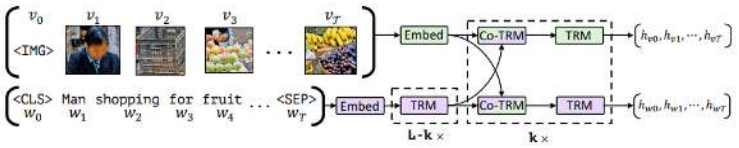
attention" module is used. **The "co-attention" module calculates importance scores based on both images and text embeddings**.



Standard encoder transformer block VS **co-attention** transformer layer

Standard self-attention VS VilBERT's proposed co-attention [4]

In a way, the model is learning the alignment between words and image regions. Another transformer module is added on top for refinement. This "co-attention" / transformer block can, of course, be repeated many times.



VilBERT processes images and text in two parallel streams that interact through co-attention [4]

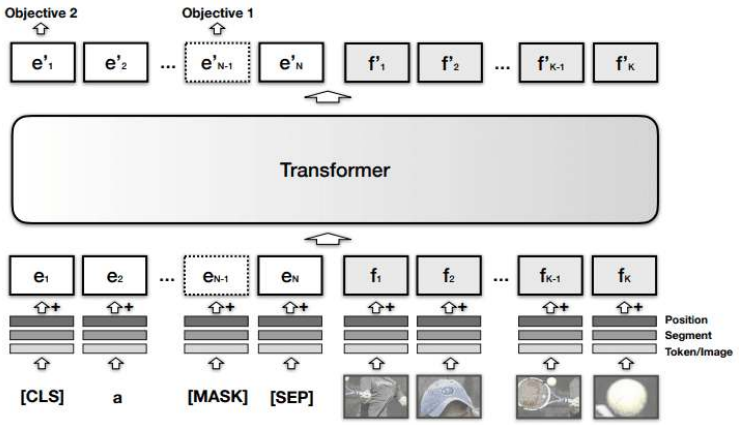
The two sides of the model are initialized separately. Regarding the text stream (**purple**), the weights are set by pretraining the model on a standard text corpus, while for the image stream (**green**), a Faster R-CNN is used. The entire model is trained on a dataset of image-text pairs with the end objective being to understand the relationship between text and images. The pretrained model can then be fine-tuned to a variety of downstream VL tasks.

Single-stream models

In contrast, models such as VisualBERT [3], VL-BERT [7], UNITER [10] **encode both modalities within the same module**. For example, VisualBERT combines image regions and language with a transformer in order for

BERT architecture. The visual embedding consists of :

1. A visual feature representation of the region produced by a CNN
2. A segment embedding that distinguishes image from text embeddings
3. A positional embedding to align regions with words if provided in the input



VisualBERT combines image regions and text with a transformer module [3]

Pretraining and fine-tuning

The performance benefits of these models are partially due to the fact that they are pretrained on huge datasets. Visual BERT-like models are usually pretrained on paired image + text datasets, learning general multimodal representations. Afterwards, they are fine-tuned on downstream tasks such as visual question answering (VQA), etc with specific datasets.

Let's explore some common pretraining strategies.

Pretraining strategies

1. **Masked Language Modeling** is often used when the transformer is trained only on text. Certain tokens of the input are being masked at random. The model is trained to simply predict the masked tokens (words). In the case of BERT, bidirectional training enables the model to use both previous and following tokens as context for prediction.
2. **Next Sequence Prediction** works again only with text

using both false and correct sentences as training data, the model is able to capture long-term dependencies.

3. **Masked Region Modeling** masks image regions in a similar way to masked language modeling. The model is then trained to predict the features of the masked region.
4. **Image-Text Matching** forces the model to predict if a sentence is appropriate for a specific image.
5. **Word-Region Alignment** finds correlations between image region and words.
6. **Masked Region Classification** predicts the object class for each masked region.
7. **Masked Region Feature Regression** learns to regress the masked image region to its visual features.

For example, VisualBERT is pretrained with the Masked Language Modeling and Image-text matching on an image-caption dataset.

The above methods create supervised learning objectives. Either the label is derived from the input, aka self-supervised or a labeled dataset (usually image-text pairs) is used. Are there any other attempts? Of course.

The following strategies are also used in VL modeling. They are often combined on various proposals.

1. **Unsupervised VL Pretraining** usually refers to pretraining without paired image-text data but rather with a single modality. During fine-tuning though, the model is fully-supervised.
2. **Multi-task Learning** is the concept of joint learning across multiple tasks in order to transfer the learnings from one task to another.
3. **Contrastive Learning** is used to learn visual-semantic embeddings in a self-supervised way. The main idea is to learn such an embedding space in which similar pairs stay close to each other while

4. **Zero-shot learning** is the ability to generalize at inference time on samples from unseen classes.

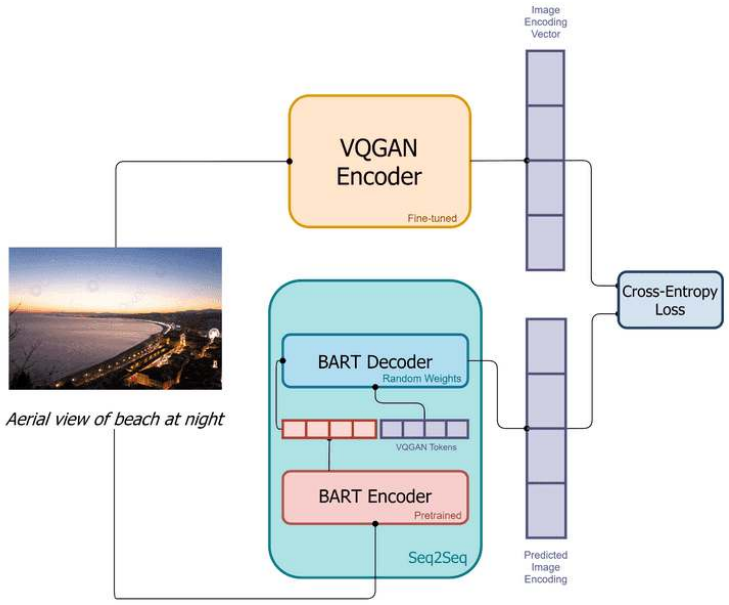
Let's now proceed with some of the most popular architectures.

VL Generative models

DALL-E

DALL-E [11] tackles the visual generation (VG) problem by being able to generate accurate images from a text description. The architecture is again trained with a text-images pair dataset.

DALL-E uses a discrete variational autoencoder (dVAE [12]) to map the images to image tokens. dVAE essentially uses a discrete latent space compared to a typical VAE. The text is tokenized with byte-pair encoding. The image and text tokens are concatenated and processed as a single data stream.



Training pipeline of DALL-E mini, slightly different from the original DALL-e

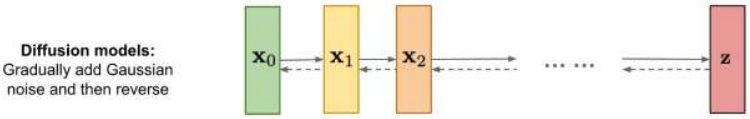
DALL-E uses an autoregressive transformer to process the stream in order to model the joint distribution of text and images. In the transformer's decoder, each image can attend to all text tokens. At inference time, we concatenate the tokenized target caption with a sample from the dVAE, and pass the data stream to the autoregressive decoder, which will output a novel token image.

DALL-E provides some exceptional results (although admittedly a little cartoonized) as you can see in the image below.

DALL-E generates realistic images based on a textual description. Source: DALL·E: Creating Images from Text

GLIDE

Following the work of DALLE, GLIDE [13] is another generative model that seems to outperform previous efforts. GLIDE is essentially a diffusion model.



Diffusion models consists of multiple diffusion steps that slowly add random noise to the data. Then, they aim to learn to reverse the diffusion process to construct samples from the data distribution from noise. Source: lilianweng

Diffusion models, in a nutshell, work by slowly injecting random noise to the data in a sequential fashion (formulated as a Markov chain). They then learn to reverse the process in order to construct novel data from the noise. So instead of sampling from the original unknown data distribution, they can sample from a known data distribution produced after a series of diffusion steps. In fact, it can be proved that if we add gaussian noise, the end (limit) distribution will be a typical normal distribution.

The diffusion model receives input as images and can output novel ones. But it can also be conditioned on textual information so that the generated image will be appropriate for specific text inputs. And that's exactly

Mathematically, the diffusion process can be formulated as follows. If we take a sample x_0 from a data distribution $q(x_0)$, we can produce a Markov chain of latent variables $x_1, \dots x_T$ by progressively adding Gaussian noise of magnitude $1 - a_t$:

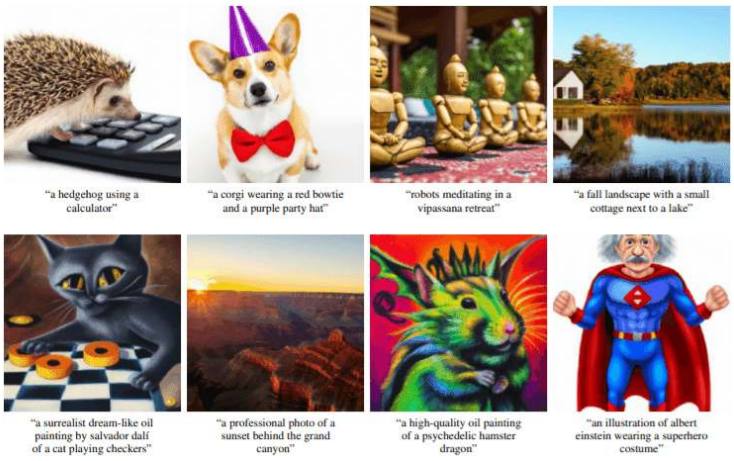
$$q(x_t|x_{t-1}) := N(x_t; \sqrt{a_t}x_{t-1}, (1 - a_t)I)$$

That way, we can well-define the posterior $q(x_{t-1}|x_t)$ and approximate it using a model $p_\theta(x_{t-1}|x_t)$.

$$p_\theta(x_{t-1}|x_t) := N(\mu_\theta(x_t), \Sigma_\theta(x_t))$$

To better understand diffusion models, I highly recommend this excellent article by Lillian Weng.

GLIDE results are even more impressive and more realistic than DALLE. However, as the authors themselves admit, there have been quite a few failure cases for specific unusual objects or scenarios. Note that you can try it yourself using hugging face spaces.



Example of generated images by GLIDE [13]

VL models based on contrastive learning

CLIP

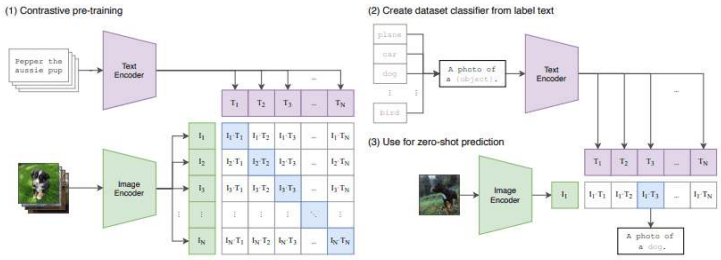
CLIP [14] targets the Natural Language for Visual Reasoning (NLVR) problem as it tries to classify an image to a specific label based on its context. The label is usually a phrase or a sentence describing the image. More interestingly, it's a zero-shot classifier in terms that it can be used to previously unseen labels.

heavily affected by the fact that it is trained on a highly-diversified, huge (400 million) dataset. The training data consist of images and their corresponding textual descriptions. The images are encoded by either a ResNet or a transformer, while a transformer module is also used for text.

The training's objective is to "connect" image representations with text representations. In a few words, the model tries to discover which text vector is more "appropriate" for a given image vector. This is why it's referred to as contrastive learning.

For those familiar with purely vision-based contrastive learning, here instead of bringing together views of the same image, we are pulling together the positive image and text "views", while pulling apart texts that do not correspond to the correct image (negatives). So even though it's contrastive training it's 100% supervised, meaning that labeled pairs are required.

By training the model to assign high similarity for fitting image-text pairs and low similarity for unfitting ones, the model can be used in a variety of downstream tasks such as image recognition.



In CLIP, the image encoder and the text encoder are trained jointly in a contrastive fashion [14]

Borrowed from the original paper, you can find a pseudocode implementation below:

```
# image_encoder - ResNet or Vision Transformer
# text_encoder - CBOW or Text Transformer

# I[n, h, w, c] - minibatch of aligned images

# T[n, l] - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t - learned temperature parameter
```

```
# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss = (loss_i + loss_t)/2
```

> The results are again quite impressive, but limitations still exist. For example, CLIP seems to struggle with abstract concepts and has **poor generalization to images not covered in its pre-training dataset**.

Example of caption prediction for n image using CLIP. Source: CLIP: Connecting Text and Images

ALIGN

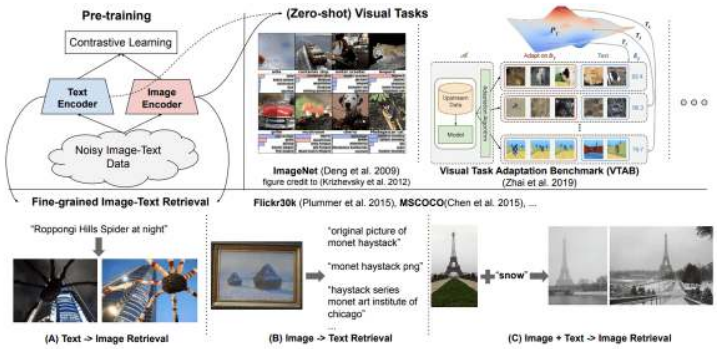
In a very similar way, ALIGN [15] utilizes a dual-encoder that learns to align visual and language representations of image-text pairs. The encoder is trained with a contrastive loss, which is formalized as a normalized softmax. In more detail, they authors use two loss terms, one for image-to-text classification and one for text-to-image classification.

Given xi and yj the normalized embedding of the image in the $i-th$ pair and that of text in the $j-th$ pair respectively, N the batch size, and σ the temperature to scale the logits, the loss functions can be defined as:

$$L_{i2t} = -\frac{1}{N}\sum_{i}^{N} log \frac{exp(x_i^T y_i/\sigma))}{\sum_{j=1}^{N} exp(x_i^T y_i/\sigma))}$$

$$L_{t2i} = -\frac{1}{N}\sum_{i}^{N} log \frac{exp(y_i^T x_i/\sigma))}{\sum^{N} exp(y^T x_i/\sigma))}$$

performed with a noisy dataset of one billion image-text pairs. So instead of doing expensive preprocessing on the data as similar methods do, they show that the scale of the dataset can compensate for the extra noise.



In ALIGN, Visual and language representation are learned jointly with contrastive learning [15]

FLORENCE

Florence [16] combines many of the aforementioned techniques to propose a new paradigm of end-to-end learning for VL tasks. The authors view Florence as a foundation model (following the terminology proposed by the Stanford team at Bommasani et al). Florence is the most recent architecture in this article and seems to perform SOTA results in many different tasks. Its main contributions include:

- For pretraining, they use a hierarchical vision transformer (Swin) as the image encoder and a modified CLIP as the language decoder.
- The training is performed on "image-label-description" triplets.
- They use a unified image-text learning scheme, which can be seen as bidirectional contrastive learning. Without diving too deep, the loss contains two contrastive terms; an image-to-language contrastive loss and a language-to-image contrastive loss. In a way, they try to combine two common learning tasks: the mapping of images to the labels and the assignment of a description to a unique label.
- They enhance the pretrained representations into

depend on the task: object-level representations, visual-language representations, video representations.

That way, the model can be applied into many distinct tasks and appears to have very good zero-shot and few-shot performance.

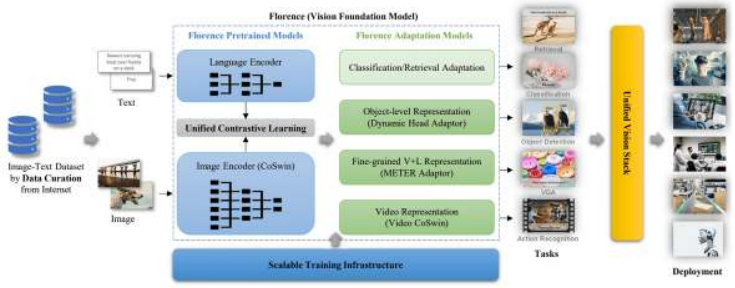


Illustration of Florence architecture [16]

Enhanced visual representations

While text encoding is usually done with a transformer-like module, visual encoding is still an area of active research. Many different proposals have been made over the years. Images have been processed with typical CNNs, ResNets, or Transformers. DALL-E even used a dVAE to compress the visual information in a discrete latent space. This is similar to words that are mapped to a discrete set of embeddings comprising the dictionary, but for image patches. Nonetheless, building better image encoding modules is a top priority at the moment.

VinVL

Towards that goal, the authors of VinVL [17] pretrained a novel model on object detection using four public datasets. They then added an "attribute" branch and fine-tuned it, making it capable of detecting both objects and attributes.

> An attribute is a small textual description related to the image.

The resulted object-attribute detection model is a modification of the Faster-RCNN model and can be used to derive accurate image representations

SimVLM [18], on the other hand, utilizes a version of the vision transformer (Vit). In fact, they replaced the well-known patch projection with three ResNet blocks to extract image patch vectors (Conv stage in the image below). The ResNet blocks are trained together with the entire model, contrary to other methods where a fully-pretrained image module is used.

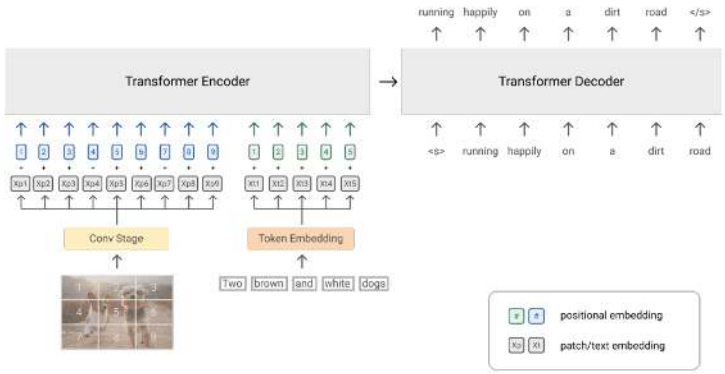


Illustration of SimVLM. The model is pretrained with a unified objective, similar to language modeling, using large-scale weakly labeled data [18]

Conclusion and observations

Given the fact that all the said models are barely new, it seems that the research community still has a long way to go in order to build solid visual language models. We have seen an explosion of very similar architectures from different teams, all following the pretraining/ fine-tune paradigm of large-scale transformers. I could include many more architectures in this article, but it seems that it wouldn't have provided much value.

The thing that concerns me is that the majority of the models come from big-tech companies, which is clearly a sign that huge datasets and infrastructure needs are required.

It is also clear to me that contrastive learning approaches are the go-to method for the moment with CLIP and ALIGN being instrumental in this direction. While the text encoding part is kind of "solved", much

effort is needed to gain better visual representations. Moreover, generative models such as DALLE and GLIDE have shown very promising results, but they also come with many limitations.

models, there are some excellent surveys that can start from [19] [20] [21] [22].

As always, thanks for your interest in our content. Community support (like social media sharing) is always appreciated. Stay tuned for more.

Cite as

```
@article{karagiannakos2022visionlanguagemodels,
    title   = "Vision Language models: towards multi-mo
    author  = "Karagiannakos, Sergios",
    journal = "https://theaisummer.com/",
    year    = "2022",
    howpublished = {https://theaisummer.com/vision-lang
  }
```

References

1. Mogadala, Aditya, et al. “Trends in Integration of Vision and Language Research: A Survey of Tasks, Datasets, and Methods.” Journal of Artificial Intelligence Research, vol. 71, Aug. 2021, pp. 1183–317↩
2. Devlin, Jacob, et al. “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding.” ArXiv:1810.04805 [Cs], May 2019↩
3. Li, Liunian Harold, et al. “VisualBERT: A Simple and Performant Baseline for Vision and Language.” ArXiv:1908.03557 [Cs], Aug. 2019↩
4. Lu, Jiasen, et al. “ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks.” ArXiv:1908.02265 [Cs], Aug. 2019↩
5. Huang, Zhicheng, et al. “Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers.” ArXiv:2004.00849 [Cs], June 2020↩
6. Qi, Di, et al. “ImageBERT: Cross-Modal Pre-Training with Large-Scale Weak-Supervised Image-Text Data.” ArXiv:2001.07966 [Cs], Jan. 2020↩
7. Su, Weijie, et al. “VL-BERT: Pre-Training of Generic Visual-Linguistic Representations.” ArXiv:1908.08530 [Cs], Feb. 2020↩

Dialog Transformer with BERT." ArXiv:2004.13278 [Cs], Nov. 2020↩

9. Tan, Hao, and Mohit Bansal. "LXMERT: Learning Cross-Modality Encoder Representations from Transformers." ArXiv:1908.07490 [Cs], Dec. 2019↩
10. Chen, Yen-Chun, et al. "UNITER: UNiversal Image-TExt Representation Learning." ArXiv:1909.11740 [Cs], July 2020↩
11. Ramesh, Aditya, et al. "Zero-Shot Text-to-Image Generation." ArXiv:2102.12092 [Cs], Feb. 2021↩
12. Rolfe, Jason Tyler. "Discrete Variational Autoencoders." ArXiv:1609.02200 [Cs, Stat], Apr. 2017↩
13. Nichol, Alex, et al. "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models." ArXiv:2112.10741 [Cs], Dec. 2021↩
14. Radford, Alec, et al. "Learning Transferable Visual Models From Natural Language Supervision." ArXiv:2103.00020 [Cs], Feb. 2021↩
15. Jia, Chao, et al. "Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision." ArXiv:2102.05918 [Cs], June 2021↩
16. Yuan, Lu, et al. "Florence: A New Foundation Model for Computer Vision." ArXiv:2111.11432 [Cs], Nov. 2021↩
17. Zhang, Pengchuan, et al. "VinVL: Revisiting Visual Representations in Vision-Language Models." ArXiv:2101.00529 [Cs], Mar. 2021↩
18. Wang, Zirui, et al. "SimVLM: Simple Visual Language Model Pretraining with Weak Supervision." ArXiv:2108.10904 [Cs], Aug. 2021↩
19. Baltrušaitis, Tadas, et al. "Multimodal Machine Learning: A Survey and Taxonomy." ArXiv:1705.09406 [Cs], Aug. 2017↩
20. Guo, Wenzhong, et al. "Deep Multimodal Representation Learning: A Survey." IEEE Access, vol. 7, 2019, pp. 63373–94. IEEE Xplore↩
21. Zhang, Chao, et al. "Multimodal Intelligence: Representation Learning, Information Fusion, and Applications." IEEE Journal of Selected Topics in Signal Processing, vol. 14, no. 3, Mar. 2020, pp. 478–

22. Uppal, Shagun, et al. "Multimodal Research in Vision and Language: A Review of Current and Emerging Trends." ArXiv:2010.09522 [Cs], Dec. 2020↩