Multimodal Deep Learning

Contents

Preface

Author: Matthias Aßenmacher

FIGURE 1: LMU seal (left) style-transferred to Van Gogh's Sunflower painting (center) and blended with the prompt - Van Gogh, sunflowers - via CLIP+VGAN (right).

In the last few years, there have been several breakthroughs in the methodologies used in Natural Language Processing (NLP) as well as Computer Vision (CV). Beyond these improvements on single-modality models, large-scale multi-modal approaches have become a very active area of research.

In this seminar, we reviewed these approaches and attempted to create a solid overview of the field, starting with the current state-of-the-art approaches in the two subfields of Deep Learning individually. Further, modeling frameworks are discussed where one modality is transformed into the other Chapter 3.1 and Chapter 3.2), as well as models in which one modality is utilized to enhance representation learning for the other (Chapter 3.3 and Chapter 3.4). To conclude the second part, architectures with a focus on handling both modalities simultaneously are introduced (Chapter 3.5). Finally, we also cover other modalities (Chapter 4.1 and Chapter 4.2) as well as general-purpose multi-modal models (Chapter 4.3), which are able to handle different tasks on different modalities within one unified architecture. One interesting application (Generative Art, Chapter 4.4) eventually caps off this booklet.

FIGURE 2: Creative Commons License

Foreword

Author: Matthias Aßenmacher

This book is the result of an experiment in university teaching. We were inspired by a group of other PhD Students around Christoph Molnar, who conducted another seminar on Interpretable Machine Learning in this format. Instead of letting every student work on a seminar paper, which more or less isolated from the other students, we wanted to foster collaboration between the students and enable them to produce a tangible outout (that isn't written to spend the rest of its time in (digital) drawers). In the summer term 2022, some Statistics, Data Science and Computer Science students signed up for our seminar entitled "Multimodal Deep Learning" and had (before kick-off meeting) no idea what they had signed up for: Having written an entire book by the end of the semester.

We were bound by the examination rules for conducting the seminar, but otherwise we could deviate from the traditional format. We deviated in several ways:

1. Each student project is a chapter of this booklet, linked contentwise to other chapers since there's partly a large overlap between the topics.
2. We gave challenges to the students, instead of papers. The challenge was to investigate a specific impactful recent model or method from the field of NLP, Computer Vision or Multimodal Learning.
3. We designed the work to live beyond the seminar.
4. We emphasized collaboration. Students wrote the introduction to chapters in teams and reviewed each others individual texts.

Technical Setup

The book chapters are written in the Markdown language. The simulations, data examples and visualizations were created with R (R Core Team, 2018). To combine R-code and Markdown, we used rmarkdown. The book was compiled

with the bookdown package. We collaborated using git and github. For details, head over to the book's repository.

1

Introduction

Author: Nadja Sauter

Supervisor: Matthias Aßenmacher

1.1 Introduction to Multimodal Deep Learning

There are five basic human senses: hearing, touch, smell, taste and sight. Possessing these five modalities, we are able to perceive and understand the world around us. Thus, "multimodal" means to combine different channels of information simultaneously to understand our surroundings. For example, when toddlers learn the word "cat", they use different modalities by saying the word out loud, pointing on cats and making sounds like "meow". Using the human learning process as a role model, artificial intelligence (AI) researchers also try to combine different modalities to train deep learning models. On a superficial level, deep learning algorithms are based on a neural network that is trained to optimize some objective which is mathematically defined via the so-called loss function. The optimization, i.e. minimizing the loss, is done via a numerical procedure called gradient descent. Consequently, deep learning models can only handle numeric input and can only result in a numeric output. However, in multimodal tasks we are often confronted with unstructured data like pictures or text. Thus, the first major problem is how to represent the input numerically. The second issue with regard to multimodal tasks is how exactly to combine different modalities. For instance, a typical task could be to train a deep learning model to generate a picture of a cat. First of all, the computer needs to understand the text input "cat" and then somehow translate this information into a specific image. Therefore, it is necessary to identify the contextual relationships between words in the text input and the spatial relationships betweent pixels in the image output. What might be easy for a toddler in pre-school, is a huge challenge for the computer. Both have to learn some understanding of the word "cat" that comprises the meaning and appearance of the animal. A common approach in modern deep learning is to generate embeddings that represent the cat numerically as a vector in some latent space. However, to achieve this, different approaches and algorithmic

architectures have been developed in recent years. This book gives an overview of the different methods used in state-of-the-art (SOTA) multimodal deep learning to overcome challenges arising from unstructured data and combining inputs of different modalities.

1.2 Outline of the Booklet

Since multimodal models often use text and images as input or output, methods of Natural Language Processing (NLP) and Computer Vision (CV) are introduced as foundation in Chapter 2. Methods in the area of NLP try to handle text data, whereas CV deals with image processing. With regard to NLP (subsection 2.1), one concept of major importance is the so-called word embedding, which is nowadays an essential part of (nearly) all multimodal deep learning architectures. This concept also sets the foundation for transformer-based models like BERT (Devlin et al., 2018a), which achieved a huge improvement in several NLP tasks. Especially the (self-)attention mechanism (Vaswani et al., 2017a) of transformers revolutionized NLP models, which is why most of them rely on the transformer as a backbone. In Computer Vision (subsection 2.2) different network architectures, namely ResNet (He et al., 2015), EfficientNet (Tan and Le, 2019a), SimCLR (Chen et al., 2020a) and BYOL (Grill et al., 2020b), will be introduced. In both fields it is of great interest to compare the different approaches and their performance on challenging benchmarks. For this reason, the last subsection 2.3 of Chapter 2 gives an overall overview of different data sets, pre-training tasks and benchmarks for CV as well as for NLP.

The second Chapter (see 3) focuses on different multimodal architectures, covering a wide variety of how text and images can be combined. The presented models combine and advance different methods of NLP and CV. First of all, looking at Img2Text tasks (subsection 3.1), the data set Microsoft COCO for object recognition (Lin et al., 2014a) and the meshed-memory transformer for Image Captioning (M^2 Transformer) (Cornia et al., 2019) will be presented. Contrariwise, researchers developed methods to generate pictures based on a short text prompt (subsection 3.2). The first models accomplishing this task were generative adversarial networks (GANs) (Goodfellow et al., 2014b) and Variational Autoencoders (VAEs) (Kingma and Welling, 2019). These methods were improved in recent years and today's SOTA transformer architectures and text-guided diffusion models like DALL-E (Ramesh et al., 2021a) and GLIDE (Nichol et al., 2021a) achieve remarkable results. Another interesting question is how images can be utilized to support language models (subsection 3.3). This can be done via sequential embeddings, more advanced grounded embeddings or, again, inside transformers. On the other hand, one can also look at text

supporting CV models like CLIP (Radford et al., 2021b), ALIGN (Jia et al., 2021a) and Florence (Yuan et al., 2021) (subsection 3.4). They use foundation models meaning reusing models (e.g. CLIP inside DALL-E 2) as well as a contrastive loss for connecting text with images. Besides, zero-shooting makes it possible to classify new and unseen data without expensive fine-tuning. Especially the open-source architecture CLIP (Radford et al., 2021b) for image classification and generation attracted a lot of attention last year. In the end of the second chapter, some further architectures to handle text and images simultaneously are introduced (subsection 3.5). For instance, Data2Vec uses the same learning method for speech, vision and language and in this way aims to find a general approach to handle different modalities in one architecture. Furthermore, VilBert (Lu et al., 2019a) extends the popular BERT architecture to handle both image and text as input by implementing co-attention. This method is also used in Google's Deepmind Flamingo (Alayrac et al., 2022). In addition, Flamingo aims to tackle multiple tasks with a single visual language model via few-shot learning and freezing the pre-trained vision and language model.

In the last chapter (see 4), methods are introduced that are also able to handle modalities other than text and image, like e.g. video, speech or tabular data. The overall goal here is to find a general multimodal architecture based on challenges rather than modalities. Therefore, one needs to handle problems of multimodal fusion and alignment and decide whether you use a join or coordinated representation (subsection 4.1). Moreover we go more into detail about how exactly to combine structured and unstructured data (subsection 4.2). Therefore, different fusion strategies which evolved in recent years will be presented. This is illustrated in this book by two use cases in survival analysis and economics. Besides this, another interesting research question is how to tackle different tasks in one so called multi-purpose model (subsection 4.3) like it is intended to be created by Google researchers (Barham et al., 2022) in their "Pathway" model. Last but not least, we show one exemplary application of Multimodal Deep Learning in the arts scene where image generation models like DALL-E (Ramesh et al., 2021a) are used to create art pieces in the area of Generative Arts (subsection 4.4).

2

Introducing the modalities

Authors: Cem Akkus, Vladana Djakovic, Christopher Benjamin Marquardt

Supervisor: Matthias Aßenmacher

Natural Language Processing (NLP) has existed for about 50 years, but it is more relevant than ever. There have been several breakthroughs in this branch of machine learning that is concerned with spoken and written language. For example, learning internal representations of words was one of the greater advances of the last decade. Word embeddings (Mikolov et al. (2013a), Bojanowski et al. (2016)) made it possible and allowed developers to encode words as dense vectors that capture their underlying semantic content. In this way, similar words are embedded close to each other in a lower-dimensional feature space. Another important challenge was solved by Encoder-decoder (also called sequence-to-sequence) architectures Sutskever et al. (2014), which made it possible to map input sequences to output sequences of different lengths. They are especially useful for complex tasks like machine translation, video captioning or question answering. This approach makes minimal assumptions on the sequence structure and can deal with different word orders and active, as well as passive voice.

A definitely significant state-of-the-art technique is Attention Bahdanau et al. (2014), which enables models to actively shift their focus – just like humans do. It allows following one thought at a time while suppressing information irrelevant to the task. As a consequence, it has been shown to significantly improve performance for tasks like machine translation. By giving the decoder access to directly look at the source, the bottleneck is avoided and at the same time, it provides a shortcut to faraway states and thus helps with the vanishing gradient problem. One of the most recent sequence data modeling techniques is Transformers (Vaswani et al. (2017b)), which are solely based on attention and do not have to process the input data sequentially (like RNNs). Therefore, the deep learning model is better in remembering context-induced earlier in long sequences. It is the dominant paradigm in NLP currently and even makes better use of GPUs, because it can perform parallel operations. Transformer architectures like BERT (Devlin et al., 2018b), T5 (Raffel et al., 2019a) or GPT-3 (Brown et al., 2020) are pre-trained on a large corpus and can be fine-tuned for specific language tasks. They have the capability to generate stories, poems, code and much more. With the help of the aforementioned

breakthroughs, deep networks have been successful in retrieving information and finding representations of semantics in the modality text. In the next paragraphs, developments for another modality image are going to be presented.

Computer vision (CV) focuses on replicating parts of the complexity of the human visual system and enabling computers to identify and process objects in images and videos in the same way that humans do. In recent years it has become one of the main and widely applied fields of computer science. However, there are still problems that are current research topics, whose solutions depend on the research's view on the topic. One of the problems is how to optimize deep convolutional neural networks for image classification. The accuracy of classification depends on width, depth and image resolution. One way to address the degradation of training accuracy is by introducing a deep residual learning framework (He et al., 2015). On the other hand, another less common method is to scale up ConvNets, to achieve better accuracy is by scaling up image resolution. Based on this observation, there was proposed a simple yet effective compound scaling method, called EfficientNets (Tan and Le, 2019a).

Another state-of-the-art trend in computer vision is learning effective visual representations without human supervision. Discriminative approaches based on contrastive learning in the latent space have recently shown great promise, achieving state-of-the-art results, but the simple framework for contrastive learning of visual representations, which is called SimCLR, outperforms previous work (Chen et al., 2020a). However, another research proposes as an alternative a simple "swapped" prediction problem where we predict the code of a view from the representation of another view. Where features are learned by Swapping Assignments between multiple Views of the same image (SwAV) (Caron et al., 2020). Further recent contrastive methods are trained by reducing the distance between representations of different augmented views of the same image ('positive pairs') and increasing the distance between representations of augmented views from different images ('negative pairs'). Bootstrap Your Own Latent (BYOL) is a new algorithm for self-supervised learning of image representatios (Grill et al., 2020b).

Self-attention-based architectures, in particular, Transformers have become the model of choice in natural language processing (NLP). Inspired by NLP successes, multiple works try combining CNN-like architectures with self-attention, some replacing the convolutions entirely. The latter models, while theoretically efficient, have not yet been scaled effectively on modern hardware accelerators due to the use of specialized attention patterns. Inspired by the Transformer scaling successes in NLP, one of the experiments is applying a standard Transformer directly to the image (Dosovitskiy et al., 2020b). Due to the widespread application of computer vision, these problems differ and are constantly being at the center of attention of more and more research.

With the rapid development in NLP and CV in recent years, it was just a question of time to merge both modalities to tackle multi-modal tasks. The

release of DALL-E 2 just hints at what one can expect from this merge in the future. DALL-E 2 is able to create photorealistic images or even art from any given text input. So it takes the information of one modality and turns it into another modality. It needs multi-modal datasets to make this possible, which are still relatively rare. This shows the importance of available data and the ability to use it even more. Nevertheless, all modalities are in need of huge datasets to pre-train their models. It's common to pre-train a model and fine-tune it afterwards for a specific task on another dataset. For example, every state-of-the-art CV model uses a classifier pre-trained on an ImageNet based dataset. The cardinality of the datasets used for CV is immense, but the datasets used for NLP are of a completely different magnitude. BERT uses the English Wikipedia and the Bookscorpus to pre-train the model. The latter consists of almost 1 billion words and 74 million sentences. The pre-training of GPT-3 is composed of five huge corpora: CommonCrawl, Books1 and Books2, Wikipedia and WebText2. Unlike language model pre-training that can leverage tremendous natural language data, vision-language tasks require high-quality image descriptions that are hard to obtain for free. Widely used pre-training datasets for VL-PTM are Microsoft Common Objects in Context (COCO), Visual Genome (VG), Conceptual Captions (CC), Flickr30k, LAION-400M and LAION-5B, which is now the biggest openly accessible image-text dataset.

Besides the importance of pre-training data, there must also be a way to test or compare the different models. A reasonable approach is to compare the performance on specific tasks, which is called benchmarking. A nice feature of benchmarks is that they allow us to compare the models to a human baseline. Different metrics are used to compare the performance of the models. Accuracy is widely used, but there are also some others. For CV the most common benchmark datasets are ImageNet, ImageNetReaL, CIFAR-10(0), OXFORD-IIIT PET, OXFORD Flower 102, COCO and Visual Task Adaptation Benchmark (VTAB). The most common benchmarks for NLP are General Language Understanding Evaluation (GLUE), SuperGLUE, SQuAD 1.1, SQuAD 2.0, SWAG, RACE, ReCoRD, and CoNLL-2003. VTAB, GLUE and SuperGLUE also provide a public leader board. Cross-modal tasks such as Visual Question Answering (VQA), Visual Commonsense Reasoning (VCR), Natural Language Visual Reasoning (NLVR), Flickr30K, COCO and Visual Entailment are common benchmarks for VL-PTM.

2.1 State-of-the-art in NLP

Author: Cem Akkus

Supervisor: Matthias Aßenmacher

2.1.1 Introduction

Natural Language Processing (NLP) exists for about 50 years, but it is more relevant than ever. There have been several breakthroughs in this branch of machine learning that is concerned with spoken and written language. In this work, the most influential ones of the last decade are going to be presented. Starting with word embeddings, which efficiently model word semantics. Encoder-decoder architectures represent another step forward by making minimal assumptions about the sequence structure. Next, the attention mechanism allows human-like focus shifting to put more emphasis on more relevant parts. Then, the transformer applies attention in its architecture to process the data non-sequentially, which boosts the performance on language tasks to exceptional levels. At last, the most influential transformer architectures are recognized before a few current topics in natural language processing are discussed.

2.1.2 Word Embeddings

As mentioned in the introduction, one of the earlier advances in NLP is learning word internal representations. Before that, a big problem with text modelling was its messiness, while machine learning algorithms undoubtedly prefer structured and well-defined fixed-length inputs. On a granular level, the models rather work with numerical than textual data. Thus, by using very basic techniques like one-hot encoding or bag-of-words, a text is converted into its equivalent vector of numbers without losing information.

In the example depicting one-hot encoding (see Figure 2.1), there are ten simple words and the dark squares indicate the only index with a non-zero value.

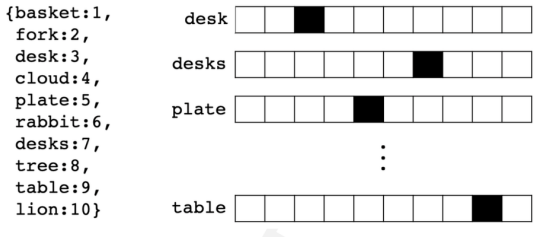


FIGURE 2.1: Ten one-hot encoded words (Source: Pilehvar and Camacho-Collados (2021))

In contrast, there are multiple non-zero values while using bag-of-words, which is another way of extracting features from text to use in modelling where we measure if a word is present from a vocabulary of known words. It is called

bag-of-words because the order is disregarded here.

Treating words as atomic units has some plausible reasons, like robustness and simplicity. It was even argued that simple models on a huge amount of data outperform complex models trained on less data. However, simple techniques are problematic for many tasks, e.g. when it comes to relevant in-domain data for automatic speech recognition. The size of high-quality transcribed speech data is often limited to just millions of words, so simply scaling up simpler models is not possible in certain situations and therefore more advanced techniques are needed. Additionally, thanks to the progress of machine learning techniques, it is realistic to train more complex models on massive amounts of data. Logically, more complex models generally outperform basic ones. Other disadvantages of classic word representations are described by the curse of dimensionality and the generalization problem. The former becomes a problem due to the growing vocabulary equivalently increasing the feature size. This results in sparse and high-dimensional vectors. The latter occurs because the similarity between words is not captured. Therefore, previously learned information cannot be used. Besides, assigning a distinct vector to each word is a limitation, which becomes especially obvious for languages with large vocabularies and many rare words.

To combat the downfalls of simple word representations, word embeddings enable to use efficient and dense representations in which similar words have a similar encoding. So words that are closer in the vector space are expected to be similar in meaning. An embedding is hereby defined as a vector of floating point values (with the length of the vector being a hyperparameter). The values for the embedding are trainable parameters which are learned similarly to a model learning the weights for a dense layer. The dimensionality of the word representations is typically much smaller than the number of words in the dictionary. For example, Mikolov et al. (2013a) called dimensions between 50-100 modest for more than a few hundred million words. For small data sets, dimensionality for the word vectors could start at 8 and go up to 1024 for larger data sets. It is expected that higher dimensions can rather pick up intricate relationships between words if given enough data to learn from.

For any NLP tasks, it is sensible to start with word embeddings because it allows to conveniently incorporate prior knowledge into the model and can be seen as a basic form of transfer learning. It is important to note that even though embeddings attempt to represent the meaning of words and do that to an extent, the semantics of the word in a given context cannot be captured. This is due to the words having static precomputed representations in traditional embedding techniques. Thus, the word "bank" can either refer to a financial institution or a river bank. Contex-

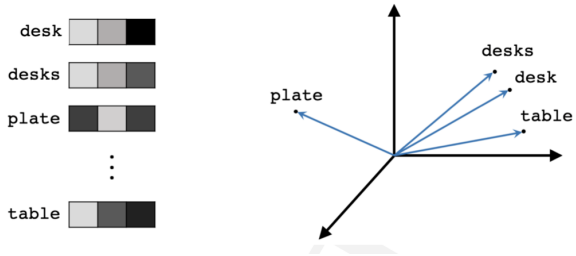


FIGURE 2.2: Three-dimensional word embeddings (Source: Pilehvar and Camacho-Collados (2021)).

tual embedding methods offer a solution, but more about them will follow later.

It should be noted that words can have various degrees of similarity. In the context of inflectional languages, it becomes obvious because words are adjusted to articulate grammatical categories. For example, in a subspace of the original vector, nouns that have similar endings can be found. However, it even exceeds simple syntactic regularities. With straightforward operations on the word vectors, it can be displayed that $vector(\text{King}) - vector(\text{Man}) + vector(\text{Woman})$ equals a vector that is closest in vector space (and therefore in meaning) to the word "Queen". A simple visualization of this relationship can be seen in the left graph below (see Figure 2.3). The three coordinate systems are representations of higher dimensions that are depicted in this way via dimension reduction techniques. Furthermore, the verb-to-tense relationship is expressed in the middle graphic, which extends the insight from before referring to the word endings being similar because in this instance the past tenses of both verbs walking and swimming are not similar in structure. Additionally, on the right side of the figure, there is a form of the commonly portrayed and easily understood Country-Capital example (see Mikolov et al. (2013a)).

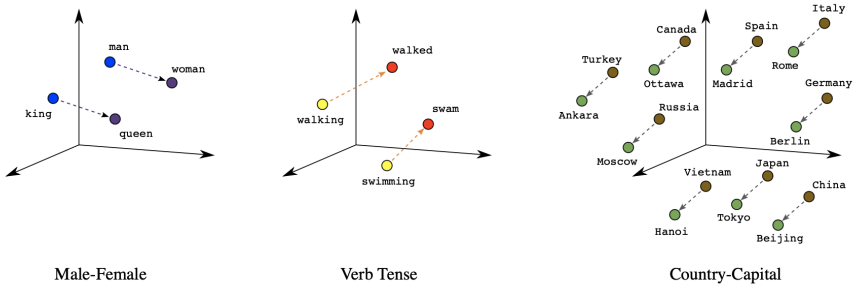


FIGURE 2.3: Three types of similarities as word embeddings (Source: Google (2022)).

Another way of using vector representations of words is in the field of translations. It has been presented that relations can be drawn from feature spaces of different languages. In below, the distributed word representations of numbers between English and Spanish are compared. In this case, the same numbers have similar geometric arrangements, which suggests that mapping linearly between vector spaces of languages is feasible. Applying this simple method for a larger set of translations in English and Spanish led to remarkable results - achieving almost 90 % precision.

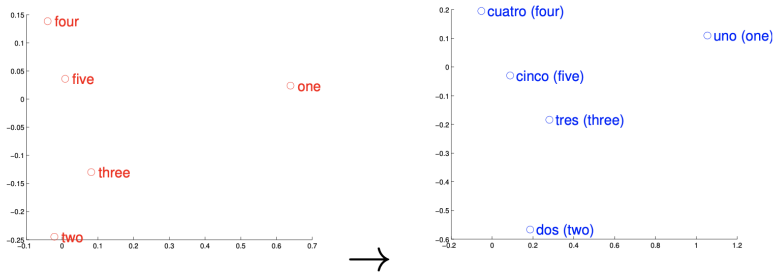


FIGURE 2.4: Representations of numbers in English and Spanish (Source: Mikolov et al. (2013c)).

This technique was then used for other experiments. One use case is the detection of dictionary errors. Taking translations from a dictionary and computing their geometric distance returns a confidence measure. Closely evaluating the translations with low confidence and outputting an alternative (one that is closest in vector space) results in a plain way to assess dictionary translations. Furthermore, training the word embeddings on a large corpora makes it possible to give sensible out-of-dictionary predictions for words. This was tested by randomly removing a part of the vocabulary before. Taking a look at the predictions revealed that they were often to some extent related to the translations with regard to meaning and semantics. Despite the accomplishments in other tasks, translations between distant languages exposed shortcomings of word embeddings. For example, the accuracy for translations between English and Vietnamese seemed significantly lower. This can be ascribed to both languages not having a good one-to-one correspondence because the concept of a word is different than in English. In addition, the used Vietnamese model contains numerous synonyms, which complicates making exact predictions (see Mikolov et al. (2013c)).

Turning the attention to one of the most impactful embedding techniques, word2vec. It was proposed by Mikolov et al. (2013a) and is not a singular algorithm. It can rather be seen as a family of model architectures and optimizations to learn word representations. Word2vec's popularity also stems from its success on multiple downstream natural language processing tasks.

It has a very simple structure which is based on a basic feed forward neural network. They published multiple papers (see Mikolov et al. (2013a)], Mikolov et al. (2013c), Mikolov et al. (2013d)) that are stemming around two different but related methods for learning word embeddings (see Figure 2.5). Firstly, the Continuous bag-of-words model aims to predict the middle word based on surrounding context words. Hence, it considers components before and after the target word. As the order of words in the context is not relevant, it is called a bag-of-words model. Secondly, the Continuous skip-gram model only considers the current word and predicts others within a range before and after it in the same sentence. Both of the models use a softmax classifier for the output layer.

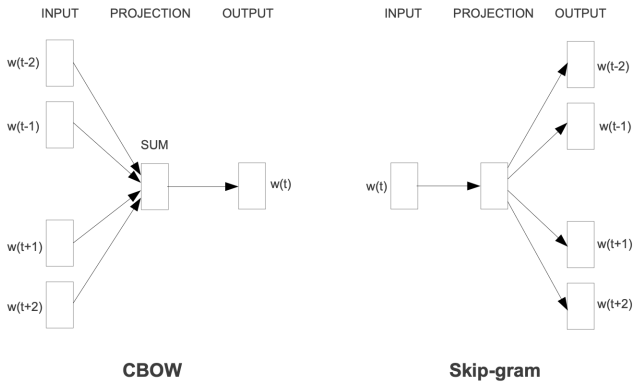


FIGURE 2.5: CBOW and Skip-gram architecture (Source: Mikolov et al. (2013a)).

Then, Bojanowski et al. (2016) built on skip-gram models by accounting for the morphology (internal structure) of words. A different classical embedding architecture that has to be at least mentioned is the GloVe model, which does not use a neural network but incorporates local context information with global co-occurrence statistics.

2.1.3 Encoder-Decoder

The field of natural language processing is concerned with a variety of different tasks surrounding text. Depending on the type of NLP problem, the network may be confronted with variable length sequences as input and/or output. This is the case for many compelling applications, such as question answering, dialogue systems or machine translation. In the following, many examples will explore machine translations in more detail, since it is a major problem domain. Regarding translation tasks, it becomes obvious that input sequences need to be mapped to output sequences of different lengths. To manage this

type of input and output, a design with two main parts could be useful. The first one is called the encoder because, in this part of the network, a variable length input sequence is transformed into a fixed state. Next, the second component called the decoder maps the encoded state to an output of a variable length sequence. As a whole, it is known as an encoder-decoder or sequence-to-sequence architecture and has become an effective and standard approach for many applications which even recurrent neural networks with gated hidden units have trouble solving successfully. Deep RNNs may have a chance, but different architectures like encoder-decoder have proven to be the most effective. It can even deal with different word orders and active, as well as passive voice (Sutskever et al., 2014). A simplified example of the encoder-decoder model can be seen in 2.6.

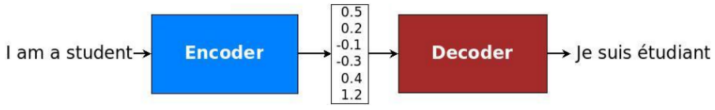


FIGURE 2.6: Translation through simplified seq2seq model (Source: Manning et al. (2022)).

Before going through the equations quantifying the concepts, it makes sense to examine the sequence-to-sequence design proposed by Cho et al. (2014). An encoder-RNN processes the input sequence of length n_x and computes a fixed-length context vector C, which is usually the final hidden state of the encoder or a simple function of the hidden states. After the input sequence is processed, it is added to the hidden state and passed forward in time through the recurrent connections between the hidden states in the encoder. Despite the context vector usually being a simple function of the last hidden state, its role cannot be underestimated. Specifically, the encoded state summarizes important information from the input sequence, e.g. the intent in a question answering task or the meaning of a text in the case of machine translation. After the context is passed to every hidden state of the decoder, the decoder RNN uses this information to produce the target sequence of length n_y, which can of course vary from n_x.

At the latest through the above illustration, it is clear that the decoder is particularly interesting to look at in the form of equations. The notation mainly follows Cho et al. (2014). The decoder is another type of RNN which is trained to predict the target based on the hidden state at the last time step. However, unlike regular RNNs, it is also conditioned on the output of the last time step (y_{t-1}) and a summary of the input c. Therefore, the hidden state of the decoder is computed by:

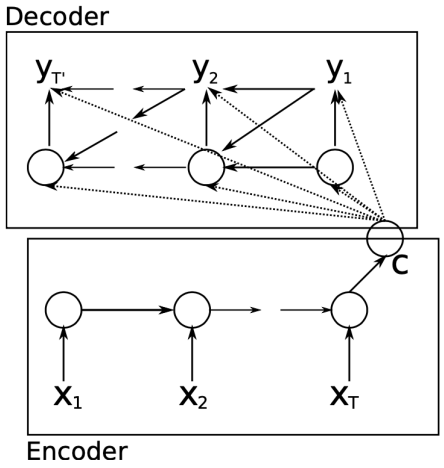


FIGURE 2.7: Encoder-decoder architecture (Source: Cho et al. (2014)).

$$h_d^{[t]} = f(h_d^{[t-1]}, y^{[t-1]}, c).$$

Similarly, each conditional probability is given by the following, where f is a non-linear activation function (and must produce probabilities in , e.g. the softmax function):

$$P(y^{[t]}|y^{[1]}, \ldots, y^{[t-1]}, c) = f(h_d^{[t]}, y^{[t-1]}, c).$$

The two parts are jointly trained to maximize the conditional log-likelihood, where θ denotes the set of model parameters and (x_n, y_n) is an (input sequence, output sequence) pair from the training set with size N:

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} \log p_\theta(y_n|x_n).$$

The best probability is usually found by using the beam search algorithm. The core idea of it is that on each step of the decoder, we keep track of the k most probable partial translations (which are called hypotheses).

Examining the translation presented in with hidden units unrolled through time could look like in 2.8. In particular, multiple hidden layers are recommended by the researchers. The idea is that lower layers compute lower-level features and higher layers compute higher-level features.

Gated recurrent networks, especially long short-term memory networks, have

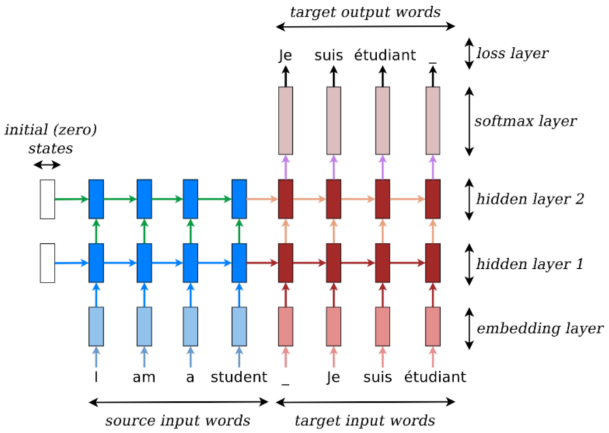


FIGURE 2.8: Translation through seq2seq model (Source: Manning et al. (2022)).

been found to be effective in both components of the sequence-to-sequence architecture. Furthermore, it was revealed that deep LSTMs significantly outperform shallow LSTMs. Each additional layer reduced perplexity by nearly 10%, possibly due to their much larger hidden state. For example, Sutskever et al. (2014) used deep LSTMs with 4 layers and 1000 cells at each layer for 1000-dimensional word embeddings. Thus, in total, 8000 real numbers are used to represent a sentence. For simplification, the neural networks are in the following referred to as RNNs which is not contradicting the insights of this paragraph as LSTMs are a type of gated RNNS (Sutskever et al., 2014).

2.1.4 Attention

Although encoder-decoder architectures simplified dealing with variable length sequences, they also caused complications. Due to their design, the encoding of the source sentence is a single vector representation (context vector). The problem is that this state must compress all information about the source sentence in a single vector and is commonly referred to as the bottleneck problem. To be precise, the entire semantics of arbitrarily long sentences need to be wrapped into a single hidden state. Moreover, it constitutes a different learning problem because the information needs to be passed between numerous time steps. This leads to vanishing gradients within the network as a consequence of factors less than 1 multiplied with each other at every point. To illustrate, the last sentence is an ideal example of one in which an encoder-decoder approach could have difficulty coping. In particular, if the sentences are longer than the ones in the training corpus (Manning et al., 2022).

Due to the aforementioned reasons, an extension to the sequence-to-sequence architecture was proposed by Bahdanau et al. (2014), which learns to align and translate jointly. For every generated word, the model scans through some positions in the source sentence where the most relevant information is located. Afterwards, based on the context around and the previously generated words, the model predicts the target word for the current time step. This approach is called attention, as it emulates human-like (cognitive) attention. As a result of directly looking at the source and bypassing the bottleneck, it provides a solution to the problem. Then, it mitigates the vanishing gradient problem, since there is now a shortcut to faraway states. Consequently, incorporating the attention mechanism has been shown to considerably boost the performance of models on NLP tasks.

A walkthrough of the example below should resolve any outstanding questions regarding the procedure of the attention mechanism. The source sentence is seen on the bottom left, which is given in French and acts as the input for the encoder-RNN (in red). Then, the attention scores (in blue) are computed by taking the dot product between the previous output word and input words. Next, the softmax function turns the scores into a probability distribution (in pink). They are used to take a weighted sum of the encoder's hidden states and form the attention output, which mostly contains information from the hidden states that received high attention. Afterwards, the attention output is concatenated with the decoder hidden state (in green), which is applied to compute the decoder output as before. In some scenarios, the attention output is also fed into the decoder (along with the usual decoder input). This specific example was chosen because "entarter" means "to hit someone with a pie" and is therefore a word that needs to be translated with many words. As a consequence of no existing direct equivalents for this phrase, it is expected that there is not only one nearly non-zero score. In this snapshot, the attention distribution can be seen to have two significant contributors.

The following equations aim to compactly represent the relations brought forward in the last paragraphs and mainly follow Manning et al. (2022). The attention scores $e^{[t]}$ are computed by scalarly combining the hidden state of the decoder with all of the hidden states of the encoder:

$$e^{[t]} = [(h_d^{[t]})^T h_e^{[1]}, \ldots, (h_d^{[t]})^T h_e^{[N]}].$$

Besides the basic dot-product attention, there are also other ways to calculate the attention scores, e.g. through multiplicative or additive attention. Although they will not be further discussed at this point, it makes sense to at least mention them. Then, applying the softmax to the scalar scores results in the attention distribution $\alpha^{[t]}$, a probability distribution whose values sum up to 1:

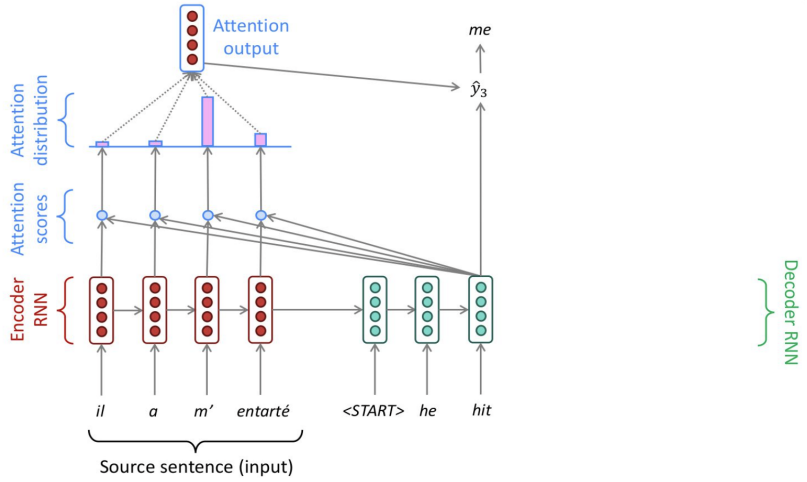


FIGURE 2.9: Translation process with attention mechanism (Source: Manning et al. (2022)).

$$\alpha^{[t]} = softmax(e^{[t]}).$$

Next, the attention output $a^{[t]}$ is obtained by the attention distribution acting as a weight for the encoder hidden states:

$$a^{[t]} = \sum_{i=1}^{N} \alpha_i^{[t]} h_{e,i}.$$

Concatenating attention output with decoder hidden state and proceeding as in the non-attention sequence-to-sequence model are the final steps:

$$o^{[t]} = f(a^{[t]} h_d^{[t]}).$$

By visualizing the attention distribution, also called alignments (see Bahdanau et al. (2014)), it is easy to observe what the decoder was focusing on and understand why it chose a specific translation. The x-axis of the plot of below corresponds to the words in the source sentence (English) and the y-axis to the words in the generated translation (French). Each pixel shows the weight of the source word for the respective target word in grayscale, where 0 is black and 1 is white. As a result, which positions in the source sentence were more relevant when generating the target word becomes apparent. As expected, the alignment between English and French is largely monotonic, as the pixels are brighter, and therefore the weights are higher along the main diagonal of the matrix. However, there is an exception because adjectives and nouns

are typically ordered differently between the two languages. Thus, the model (correctly) translated "European Economic Area" into "zone économique européene". By jumping over two words ("European" and "Economic"), it aligned "zone" with "area". Then, it looked one word back twice to perfect the phrase "zone économique européene". Additional qualitative analysis has shown that the model alignments are predominantly analogous to our intuition.

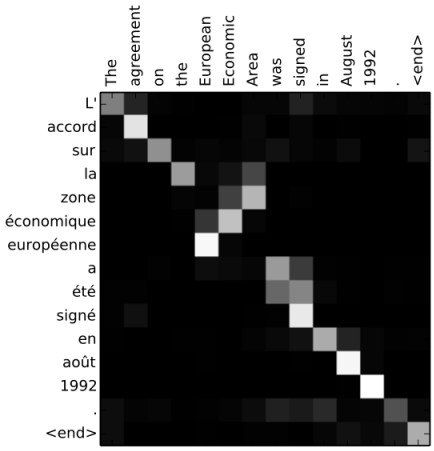


FIGURE 2.10: Attention alignments (Source: Bahdanau et al. (2014)).

2.1.5 Transformer

For this section, Manning et al. (2022) constitutes the main source.

RNNs are unrolled from one side to the other. Thus, from left to right and right to left. This encodes linear locality, which is a useful heuristic because nearby words often affect each other's meaning. But how is it when distant words need to interact with each other? For instance, if we mention a person at the beginning of a text portion and refer back to them only at the very end, the whole text in between needs to be tracked back (see below). Hence, RNNs take O(sequence length) steps for distant word pairs to interact. Due to gradient problems, it is therefore hard to learn long-distance dependencies. In addition, the linear order is ingrained. Even though, as known, the sequential structure does not tell the whole story.

GPUs can perform multiple calculations simultaneously and could help to reduce the execution time of the deep learning algorithm massively. However, forward and backward passes lack parallelizability in recurrent models and have O(sequence length). To be precise, future hidden states cannot be computed in full before past states have been computed. This inhibits training on massive

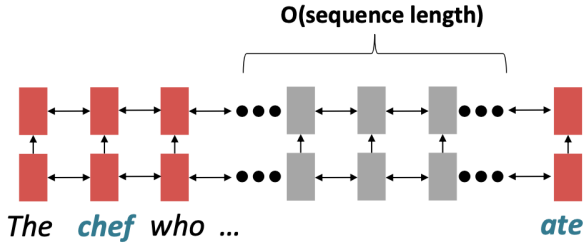


FIGURE 2.11: Sequential processing of recurrent model (Source: Manning et al. (2022)).

data sets. indicates the minimum number of steps before the respective state can be calculated.

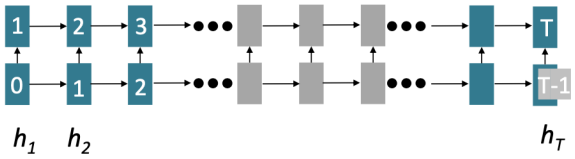


FIGURE 2.12: Sequential processing of recurrent model with number of steps indicated (Source: Manning et al. (2022)).

After proving that attention dramatically increases performance, google researchers took it further and based transformers solely on attention, so without any RNNs. For this reason, the paper in which they were introduced is called "Attention is all you need". Spoiler: It is not quite all we need, but more about that on the following pages. Transformers have achieved great results on multiple settings such as machine translation and document generation. Their parallelizability allows for efficient pretraining and leads them to be the standard model architecture. In fact, all top models on the popular aggregate benchmark GLUE are pretrained and Transformer-based. Moreover, they have even shown promise outside of NLP, e.g. in Image Classification, Protein Folding and ML for Systems (see Dosovitskiy et al. (2020a), Jumper et al. (2021), Zhou et al. (2020), respectively).

If recurrence has its flaws, another adjustment of the attention mechanism might be beneficial. Until now, it was defined from decoder to encoder. Alternatively, attention could also be from one state to all states in the same set. This is the definition of self-attention, which is encoder-encoder or decoder-decoder attention (instead of encoder-decoder) and represents a cornerstone of the transformer architecture. depicts this process in which each word attends to all words in the previous layer. Even though in practice, most

arrows are omitted eventually.

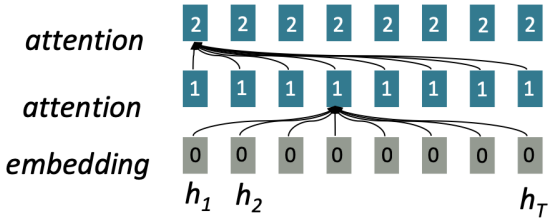


FIGURE 2.13: Connections of classic attention mechanism (Source: Manning et al. (2022)).

Thinking of self-attention as an approximate hash table eases understanding its intuition. To look up a value, queries are compared against keys in a table. In a hash table, which is shown on the left side of , there is exactly one key-value pair for each query (hash). In contrast, in self-attention, each key is matched to varying degrees by each query. Thus, a sum of values weighted by the query-key match is returned.

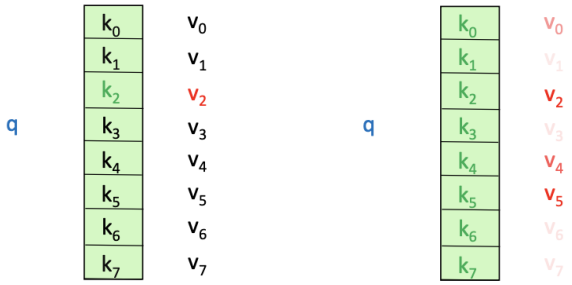


FIGURE 2.14: Comparison of classic attention mechanism with self-attention with hash tables (Source: Manning et al. (2022)).

The process briefly described in the last paragraph can be summarized by the following steps that mainly follow Manning et al. (2022). Firstly, deriving query, key, and value for each word x_i is necessary:

$$q_i = W^Q x_i, \quad k_i = W^K x_i, \quad v_i = W^V x_i$$

Secondly, the attention scores have to be calculated:

$$e_{ij} = q_i k_j$$

Thirdly, to normalize the attention scores, the softmax function is applied:

$$\alpha_{ij} = \text{softmax}(e_{ij}) = \frac{\exp(e_{ij})}{\sum_k e_{ij}}$$

Lastly, taking the weighted sum of the values results in obtaining the attention output:

$$a_i = \sum_j \alpha_{ij} v_j$$

Multiple advantages of incorporating self-attention instead of recurrences have been revealed. Since all words interact at every layer, the maximum interaction distance is $O(1)$ and is a crucial upgrade. In addition, the model is deeply bidirectional because each word attends to the context in both directions. As a result of these advances, all word representations per layer can be computed in parallel. Nevertheless, some issues have to be discussed. Attention does no more than weighted averaging. So without neural networks, there are no element-wise non-linearities. Their importance cannot be understated and shows why attention is not actually all that is needed. Furthermore, bidirectionality is not always desired. In language modelling, the model should specifically be not allowed to simply look ahead and observe more than the objective allows. Moreover, the word order is no longer encoder, and it is bag-of-words once again.

Fortunately, the previously mentioned weaknesses have been addressed for the original transformer-architecture proposed by Vaswani et al. (2017c). The first problem can be easily fixed by applying a feed forward layer to the output of attention. It provides non-linear activation as well as extra expressive power. Then, for cases in which bidirectionality contradicts the learning objective, future states can be masked so that attention is restricted to previous states. Moreover, the loss of the word can be corrected by adding position representations to the inputs.

The more complex deep learning models are, the closer they become to model the complexity of the real world. That is why the transformer encoder and decoder consist of many layers of self-attention with a feed forward network, which is necessary to extract both syntactic and semantic features from sentences. Otherwise, using word embeddings, which are semantically deep representations between words, would be unnecessary (Sejnowski, 2020). At the same time, training deep networks can be troublesome. Therefore, some tricks are applied to help with the training process.

One of them is to pass the "raw" embeddings directly to the next layer, which

prevents forgetting or misrepresent important information as it is passed through many layers. This process is called residual connections and is also believed to smoothen the loss landscape. Additionally, it is problematic to train the parameters of a given layer when its inputs keep shifting because of layers beneath. Reducing uninformative variation by normalizing within each layer to mean zero and standard deviation to one weakens this effect. Another challenge is caused by the dot product tending to take on extreme values because of the variance scaling with increasing dimensionality d_k. It is solved by Scaled Dot Product Attention (see Figure 2.15), which consists of computing the dot products of the query with its keys, dividing them by the dimension of keys $\sqrt{d_k}$, and applying the softmax function next to receive the weights of the values.

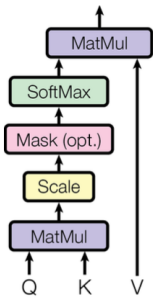


FIGURE 2.15: Scaled dot-product attention (Source: Vaswani et al. (2017c)).

Attention learns where to search for relevant information. Surely, attending to different types of information in a sentence at once delivers even more promising results. To implement this, the idea is to have multiple attention heads per layer. While one attention head might learn to attend to tense information, another might learn to attend to relevant topics. Thus, each head focuses on separate features, and construct value vectors differently. Multi-headed self-attention is implemented by simply creating n independent attention mechanisms and combining their outputs.

At this point, every part that constitutes the encoder in the transformer architecture has been introduced (see Figure 2.17). First, positional encodings are included in the input embeddings. There are multiple options to realize this step, e.g. through sinusoids. The multi-head attention follows, which was just mentioned. "Add & Norm" stands for the residual connections and the normalization layer. A feed forward network follows, which is also accompanied by residual connections and a normalization layer. All of it is repeated n times. For the decoder, the individual components are similar. One difference is that the outputs go through masked multi-head attention before multi-head attention and the feed forward network (with residual connections and layer normalization). It is critical to ensure that the decoder cannot peek at the

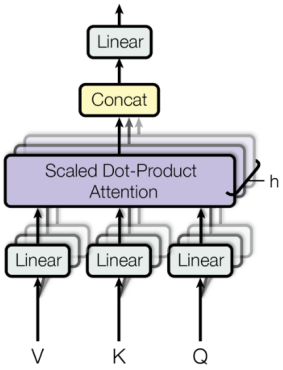


FIGURE 2.16: Multi-head attention (Source: Vaswani et al. (2017c)).

future. To execute this, the set of keys and queries could be modified at every time step to only include past words. However, it would be very inefficient. Instead, to enable parallelization, future states are masked by setting the attention scores to $-\infty$. After the decoder process is also repeated n times, a linear layer is added to project the embeddings into a larger vector that has the length of the vocabulary size. At last, a softmax layer generates a probability distribution over the possible words.

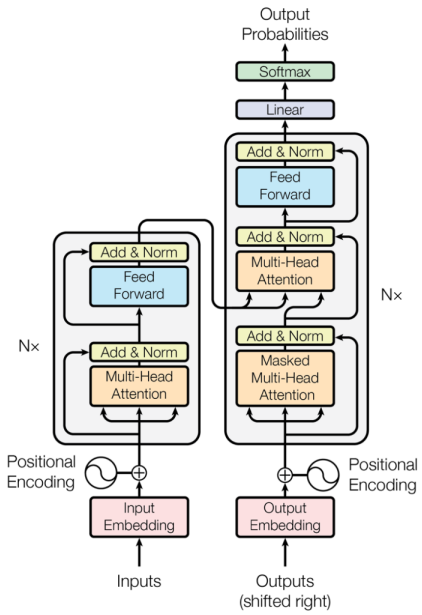


FIGURE 2.17: Transformer architecture (Source: Vaswani et al. (2017c)).

2.1.6 Transformer architectures: BERT, T5, GPT-3

"You shall know a word by the company it keeps", an adage by linguist John Rupert Firth from 1957 goes. Even earlier, in 1935, he stated that "... the complete meaning of a word is always contextual, and no study of meaning apart from a complete context can be taken seriously". The quotes of the famous linguist sum up the motivation to learn word meaning and context perfectly. Many years later, in 2017, pretraining word embeddings started. However, some complications arise from solely pretraining the first part of the network. For instance, to teach the model all contextual aspects of language, the training data for the downstream task (e.g. question answering) needs to be adequate. Additionally, most of the parameters are usually randomly initialized. presents the network discussed, in which the word "movie" gets the same embedding irrespective of the sentence it appears in. On the contrary, parameters in modern NLP architectures are initialized via pretraining (see Figure 2.18). Furthermore, during the pretraining, certain input parts are hidden to train the model to reconstruct them. This leads to building suitable parameter initializations and robust probability distributions over language.

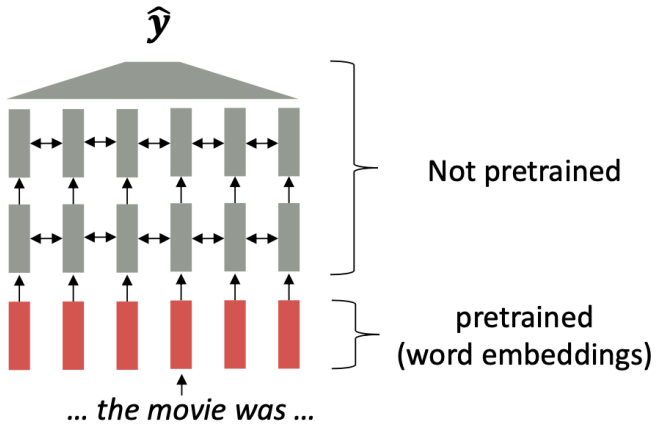


FIGURE 2.18: Partly pre-trained model (Source: Manning et al. (2022)).

Classic machine learning does not match human learning. Specifically referring to training a model from scratch, and only being able to learn from the training data. In contrast, human beings already have prior knowledge they can apply to new tasks. Transfer learning emulates this by using an already trained network. The main idea is to use a model that was pretrained on a hard, general language understanding task using endless amounts of data, so that, it eventually contains the best possible approximation of language understanding. Afterwards, the training data for the new task is applied to slightly modify the weights of

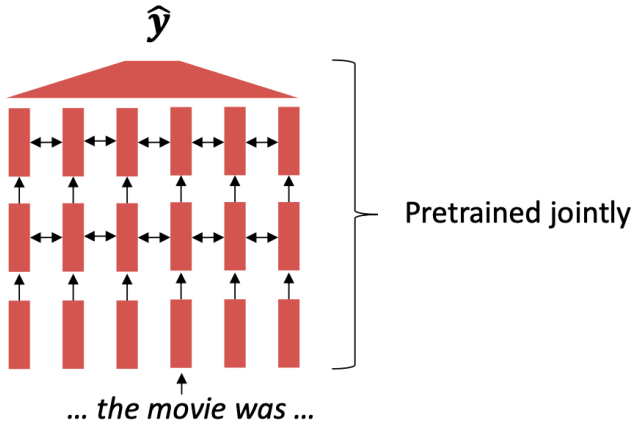


FIGURE 2.19: Jointly pre-trained model (Source: Manning et al. (2022)).

the pretrained model, which is referred to as fine-tuning (Manning et al., 2022).

The specific architecture of a transformer model affects the type of pre-training, and favourable use cases. In the following, three different but very influential transformer architectures will be discussed. BERT can be seen as stacked encoders (Devlin et al., 2018b), T5 aims to combine the good parts of encoders and decoders (Raffel et al., 2019a), while GPT are stacked decoders (Brown et al., 2020).

2.1.6.1 BERT

Transfer learning led to state-of-the-art results in natural language processing. One of the architectures that led the way was BERT, which stands for Bidirectional Encoder Representations from Transformers. It receives bidirectional context, which is why it is not a natural fit for language modelling. To train it on this objective regardless, masked language modelling was proposed. The main idea is to cover up a fraction of the input words and let the model predict them. In this way, the LM objective can be used while sustaining connections to words in the future. The masked LM for BERT randomly predicts 15% of all word tokens in each sequence. Of those, 80% are replaced by the `MASK` token, 10% by a random token, and 10% remain unchanged. Moreover, because the masked words are not even seen in the fine-tuning phase, the model cannot get complacent and relies on strong representations of non-masked words. Initially, BERT had an additional objective of whether one sentence follows another, which is known as next sentence prediction. However, it was dropped in later work due to having an insignificant effect.

BERT is hugely versatile and was greatly popular after its release. Fine-tuning BERT led to outstanding results on a variety of applications, including question answering, sentiment analysis and text summarization. Thanks to its design, if the task involves generating sequences, pretrained decoders outperform pretrained encoders like BERT. Even though, it would not be recommended for autoregressive generation, up to this day, "small" models like BERT are applied as general tools for numerous tasks.

2.1.6.2 T5

The Text-To-Text Transfer Transformer (T5) is a new model that can be regarded as an application of the insights gathered by an extensive empirical study searching for the best transfer learning techniques. It is pretrained on Colossal Clean Crawled Corpus (C4), an open-source dataset. Raffel et al. (2019a) found that the best pretraining objective to use for the encoder component was span corruption. In short, different length word groups (spans) are replaced with unique placeholders, and let the model decode them. Text preprocessing is necessary for its implementation. For the decoder, it is still a language modelling task. Compared to models like BERT, which can only output a span of the input or a class label, T5 reframes all NLP tasks into a unified text-to-text format, where inputs and outputs always consist of text strings. As a result, the same model, loss function, and hyperparameters can be used on any NLP task, such as machine translation, document summarization, question answering, and classification tasks like sentiment analysis. T5 can even be applied to regression tasks by training it to predict the string representation of a number (and not the number itself). Examples of potential use cases are depicted in below.

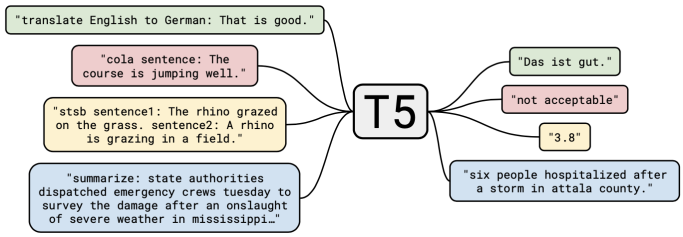


FIGURE 2.20: Applications of T5 model (Source: Raffel et al. (2019a)).

2.1.6.3 GPT-3

As previously stated, the neural architecture influences the type of pretraining. The original GPT architecture consists of a Transformer decoder with 12 layers (Radford et al., 2018). For decoders, it is sensible to simply pretrain them as language models. Afterwards, they can be used as generators to fine-tune their probability of predicting the next word conditioned on the previous words.

The models are suitable for tasks similar to the training, including any type of dialogue and document summarization. Transformer language models are great for transfer learning. They are fine-tuned by randomly initializing a softmax classifier on top of the pretrained model and training both (with only a very small learning rate and a small number of epochs) so that the gradient propagates through the whole network.

The success of BERT in 2018 prompted a "gold rush" in NLP, in which ever greater language models were created. One that topped the headlines and used a customer supercluster for computation was the third iteration of the GPT architecture by OpenAI, known as GPT-3. reveals why GPT-3 is a famous example of current research focusing on scaling up neural language models. While the largest T5 model has 11 billion parameters, GPT-3 has 175 billion parameters. Moreover, the training data set contains around 500 billion tokens of text, while the average young american child hears around 6 million words per year (Hart and Risley, 1995). The results of huge language models suggest that they perform some form of learning (without gradient steps) simply from examples provided via context. The tasks are specified by the in-context examples, and the conditional probability distribution simulates performing the task to an extent.

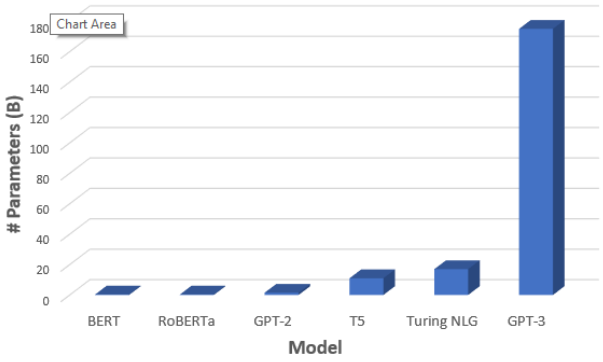


FIGURE 2.21: Comparison of number of parameters between Transformer-architectures (Source: Saifee (2020)).

2.1.7 Current Topics

2.1.7.1 Concerns regarding growing size of Language Models

As the last chapter ended with GPT-3 and emphasized the concerning trend of ever larger language models, one could ask which other costs arise from the developments. Risks and harms among environmental and financial costs have been studied by Bender et al. (2021). They state that marginalized

communities are not only less likely to benefit from LM progress, but also more likely to suffer from the environmental repercussions of increasing resource consumption. Strubell et al. (2019a) estimated that training a Transformer (big) model resulted in 249t of CO_2. To compare, an average human is responsible for approximately 5t of CO_2 per year (Ritchie et al., 2020). In addition, they discovered that an estimated increase of 0.1 in BLEU score increased computation costs by $ 150,000 (for English to German translations). Furthermore, larger models require more data to sufficiently train them. This has resulted in large but poorly documented training data sets. Multiple risks can be mitigated if there is a common understanding of the model's learnings.

Moreover, it has been argued that datasets consisting of web data over-represent hegemonic views and encode bias towards marginalized communities. This is among other factors due to internet access being unevenly distributed. In particular, there is an over-representation of younger internet users and those from developed countries. It is generally naive to educate AI systems on all aspects of the complex world, and hope for the beautiful to prevail (Bender et al., 2021).

2.1.7.2 Improving Understanding of Transformer-based models

The results of transformer-based models clearly show that they deliver successful results. However, it is less clear why. The size of the models makes it difficult to experiment with them. Nevertheless, having a limited understanding restrains researchers from coming up with further improvements. Therefore, multiple papers analysed BERT's attention in search of an improved understanding of large transformer models. BERT is a smaller model out of the more popular ones, and its attention is naturally interpretable because the attention weight indicates how significant a word is for the next representation of the current word (Clark et al., 2019). In the following, some of the findings are going to be shared.

BERT representations are rather hierarchical than linear, and they include information about parts of speech, syntactic chunks and roles (Lin et al., 2019, Liu et al. (2019a)) Furthermore, it has semantic knowledge. For example, BERT can recognize e.g. that "to tip a chef" is better than "to tip a robin" but worse than "to tip a waiter" ((Ettinger, 2019)). However, it makes sense that BERT has issues with knowledge that is assumed and not mentioned, which especially refers to visual and perceptual properties (Da and Kasai, 2019). Additionally, BERT struggles with inferences, e.g. even though it is known that "people walk into houses" and "houses are big", it cannot infer that "houses are bigger than people" (Forbes et al., 2019).

While it is true that different transformer heads attend to various patterns (see

), interestingly, most of them could be neglected without notable performance loss (Voita et al., 2019). Probing attention maps can be tedious, but allows to gain knowledge of common patterns, such as an unexpected amount focusing on the delimiter token

$$SEP$$

.

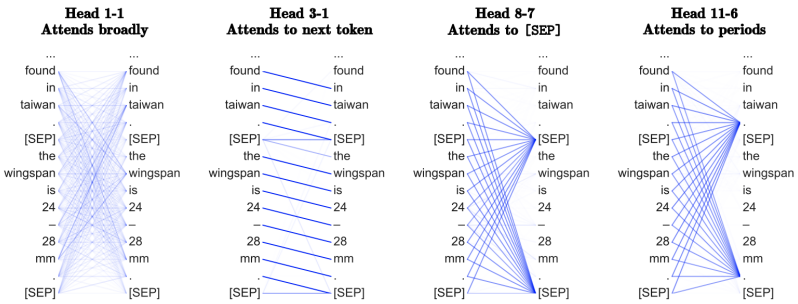


FIGURE 2.22: Common patterns of attention heads (Source: Clark et al. (2019)).

2.1.7.3 Few-Shot Learning

For NLP tasks, the model is usually trained on a set of labelled examples and is expected to generalize to unseen data. Annotating is not only costly but also difficult to gather for numerous languages, domains, and tasks. In practice, there is often only a very limited amount of labelled examples. Consequently, few-shot learning is a highly relevant research area (Schick and Schütze, 2020). It defines a model that is trained on a limited number of demonstrations to guide its predictions. Referring back to , the benefits of lower computational and environmental costs have to be mentioned.

Traditional fine-tuning uses a large corpus of example tasks, and the model is updated repeatedly with gradient steps so that it adapts to the task with minimal accuracy error.

In contrast, few-shot applications have to complete tasks at test time with only forward passes. They have three main parts: the task description, examples, and the prompt. In Figure **??**, the task is a translation from English to French, a few examples, as well as the word that should be translated are given. Moreover, zero-shot and one-shot learning refer to the model predicting with no and one learned example, respectively (Brown et al., 2020).

It is complicated to create the few-shot examples, since the application relies on them to express the task. This is why smaller models are susceptible to

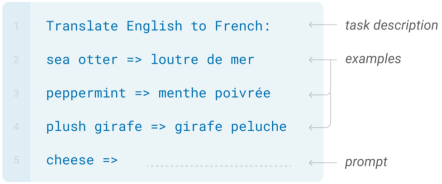


FIGURE 2.23: Few-shot learning (Source: Brown et al. (2020)).

examples written unfavourably. In Brown et al. (2020), it was shown that few-shot performance scales with the number of model parameters. Even though GPT-3's in-context learning improved few-shot prompting capabilities, it is still sensitive to the order of training examples, decoding strategy, and hyperparameter selection. All of this combined with the fact that current research uses larger or held-out data sets leads to the suspicion that the true few-shot ability of language models is overestimated (Perez et al., 2021a).

Moreover, Lialin et al. (2022) have found that common transformer models could not resolve compositional questions in a zero-shot fashion and that the model's parameter count does not correlate with performance. This indicates a limitation for zero-shot prompting with the existing pre-training objectives. However, different models provided the best accuracy with regard to different symbolic reasoning tasks. This suggests that optimization or masking strategies could be more significant than the pre-training, data set size or model architecture.

2.1.8 Summary

Natural Language Processing has been one of the most exciting fields of machine learning in the last decade considering all the breakthroughs discussed in this work. Word embeddings made it possible and allowed developers to encode words as dense vectors that capture their underlying semantic content. In this way, similar words are embedded close to each other in a lower-dimensional feature space. Another important challenge was solved by encoder-decoder (also called sequence-to-sequence) architectures, which made it possible to map input sequences to output sequences of different lengths. They are especially useful for complex tasks like machine translation, video captioning or question answering. A significant state-of-the-art technique is attention, which enabled models to actively shift their focus – just like humans do. It allows following one thought at a time while suppressing information irrelevant to the task. As a consequence, it has been shown to significantly improve performance for tasks like machine translation. By giving the decoder access to directly look at the source, the bottleneck is avoided and at the same time, it provides a shortcut to faraway states and thus helps with the vanishing gradient problem.

One of the most recent data modelling techniques is the transformer, which is solely based on attention and does not have to process the input data sequentially. Therefore, the deep learning model is better in remembering context-induced earlier in long sequences. It is the dominant paradigm in NLP currently and makes better use of GPUs because it can perform parallel operations. Transformer architectures like BERT, T5 or GPT-3 are pre-trained on a large corpus and can be fine-tuned for specific language tasks. They can generate stories, poems, code and much more. Currently, there seems to be breaking transformer news nearly every week with no sign of slowing. This is why many trends could be recognized as relevant current topics. One of them is increasing concerns regarding the growing size of language models and the correlated environmental and financial costs. Another active research aspect is concerned with improving the understanding of transformer-based models to further advance them. Additionally, there are many studies about achieving respectable results on language modelling tasks after only learning from a few examples, which is known as few-shot learning.

2.2 State-of-the-art in Computer Vision

Author: Vladana Djakovic

Supervisor: Daniel Schalk

2.2.1 History

The first research about visual perception comes from neurophysiological research performed in the 1950s and 1960s on cats. The researchers used cats as a model to understand how human vision is compounded. Scientists concluded that human vision is hierarchical and neurons detect simple features like edges followed by more complex features like shapes and even more complex visual representations. Inspired by this knowledge, computer scientists focused on recreating human neurological structures.

At around the same time, as computers became more advanced, computer scientists worked on imitating human neurons' behavior and simulating a hypothetical neural network. In his book "The Organization of Behaviour" (1949) Donald Hebbian stated that neural pathways strengthen over each successive use, especially between neurons that tend to fire at the same time, thus beginning the long journey towards quantifying the complex processes of the brain. The first Hebbian network, inspired by this neurological research, was successfully implemented at MIT in 1954 (Jaspreet, 2019).

New findings led to the establishment of the field of artificial intelligence in

1956 on-campus at Dartmouth College. Scientists began to develop ideas and research how to create techniques that would imitate the human eye.

In 1959 early research on developing neural networks was performed at Stanford University, where models called "ADALINE" and "MADALINE," (Multiple ADAptive LINear Elements) were developed. Those models aimed to recognize binary patterns and could predict the next bit (his, 2022).

Starting optimism about Computer Vision and neural networks disappeared after 1969 and the publication of the book "Perceptrons" by Marvin Minsky, founder of the MIT AI Lab, stated that the single perception approach to neural networks could not be translated effectively into multi-layered neural networks. The period that followed was known as AI Winter, which lasted until 2010, when the technological development of computer and the internet became widely used. In 2012 breakthroughs in Computer Vision happened at the ImageNet Large Scale Visual Recognition Challenge (ILSVEC). The team from the University of Toronto issued a deep neural network called AlexNet (Krizhevsky et al., 2012a) that changed the field of artificial intelligent and Computer Vision (CV). AlexNet achieved an error rate of 16.4%.

From then until today, Computer Vision has been one of the fastest developing fields. Researchers are competing to develop a model that would be the most similar to the human eye and help humans in their everyday life. In this chapter the author will describe only a few recent state-of-the-art models.

2.2.2 Supervised and unsupervised learning

As part of artificial intelligence (AI) and machine learning (ML), there are two basic approaches:

- supervised learning;
- unsupervised learning.

Supervised learning (Education, 2020a) is used to train algorithms on labeled datasets that accurately classify data or predict outcomes. With labeled data, the model can measure its accuracy and learn over time. Among others, we can distinguish between two common supervised learning problems:

- classification,
- regression.

In unsupervised learning (Education, 2020b), unlabelled datasets are analyzed and clustered using machine learning algorithms. These algorithms aim to discover hidden patterns or data groupings without previous human intervention. The ability to find similarities and differences in information is mainly used for three main tasks:

- clustering,
- association,

- dimensionality reduction.

Solving the problems where the dataset can be both labeled and unlabeled requires a semi-supervised approach that lies between supervised and unsupervised learning. It is useful when extracting relevant features from complex and high volume data, i.e., medical images.

Nowadays, a new research topic appeared in the machine learning community, Self-Supervised Learning. Self-Supervised learning is a process where the model trains itself to learn one part of the input from another (techslang, 2020). As a subset of unsupervised learning, it involves machines labeling, categorizing, and analyzing information independently and drawing conclusions based on connections and correlations. It can also be considered as an autonomous form of supervised learning since it does not require human input to label data. Unlike unsupervised learning, self-supervised learning does not focus on clustering nor grouping (Shah, 2022). One part of Self-Supervised learning is contrastive learning, which is used to learn the general features of an unlabeled dataset identifying similar and dissimilar data points. It is utilized to train the model to learn about our data without any annotations or labels (Tiu, 2021).

2.2.3 Scaling networks

Ever since the introduction of AlexNet in 2012, the problem of scaling convolutional neural networks (ConvNet) has become the topic of active research. ConvNet can be scaled in all three dimensions: depth, width, or image size. One of the first researches in 2015 showed that network depth is crucial for image classification. The question whether stacking more layers enables the network to learn better leads to deep residual networks called ResNet (He et al., 2015), which will be described in this work. Later on, scaling networks by their depth became the most popular way to improve their performance. The second solution was to scale ConvNets by their width. Wider networks tend to be able to capture more fine-grained features and are easier to train (Zagoruyko and Komodakis, 2016). Lastly, scaling the image's resolution can improve the network's performance. With higher resolution input images, ConvNets could capture more fine-grained patterns. GPipe (Huang et al., 2018) is one of the most famous networks created by this technique. The question of possibility of scaling by all three dimensions was answered by Tan and Le (2019a) in the work presenting Efficient Net. This network was built by scaling up ConvNets by all three dimensions and will also be described here.

2.2.4 Deep residual networks

The deep residual networks, called ResNets (He et al., 2015), were presented as the answer on the question whether stacking more layers would enable network to learn better. Until then one obstacle for simply stacking layers was the problem of vanishing/exploding gradients. It has been primarily addressed by

normalized initialization and intermediate normalization layers. That enabled networks with tens of layers to start converging for stochastic gradient descent (SGD) with backpropagation.

Another obstacle was a degradation problem. It occurs when the network depth increases, followed by saturating and then rapidly decreasing accuracy. Overfitting is not caused by such degradation, and adding more layers to a suitably deep model leads to higher training error, which indicates that not all systems are similarly easy to optimize.

For example, it was suggested to consider a shallower architecture and its deeper counterpart that adds more layers. One way to avoid the degradation problem is to create a deeper model, where the auxiliary layers are identity mappings and other layers are copied from a shallower model. The deeper model should produce no higher training error than its shallower counterpart. However, in practice it is not the case and it is hard to find comparably good constructs or better solutions. The solution to this degradation problem proposed by them is a deep residual learning framework.

2.2.4.1 Deep Residual Learning

2.2.4.1.1 Residual Learning

The idea of residual learning is to replace the approximation of underlying mapping $H(x)$, which is approximated by a few stacked layers (not necessarily the entire net), with an approximation of residual function $F(x) := H(x) - x$. Here x denotes the inputs to the first of these layers, and it is assumed that both inputs and outputs have the same dimensions. The original function changes its form $F(x) + x$.

A counter-intuitive phenomenon about degradation motivated this reformulation. The new deeper model should not have a more significant training error when compared to a construction using identity mappings. However, due to the degradation problem, solvers may have challenges approximating identity mappings by multiple non-linear layers. Using the residual learning reformulation can drive the weights of the non-linear layers toward zero to approach identity mappings if they are optimal. Generally, identity mappings are not optimal, but new reformulations may help to pre-condition the problem. When an optimal function is closer to an identity mapping than a zero mapping, finding perturbations concerning an identity mapping should be easier than learning the function from scratch.

2.2.4.1.2 Identity Mapping by Shortcuts

Residual learning is adopted to every few stacked layers where a building block is defined:

$$y = F(x, \{W_i\}) + x \tag{2.1}$$

x and y present the input and output vectors of the layers. Figure 2.24 visualizes the building block.

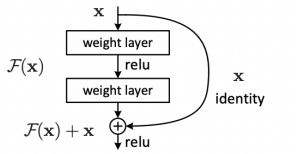


FIGURE 2.24: Building block of residual learning (He et al., 2015).

The function $F(x, \{W_i\})$ represents the residual mapping that is to be learned. For the example with two layers from Figure 2.24, $F = W_2\sigma(W_1x)$ in which σ denotes the ReLU activation function. Biases are left out to simplify the notation. The operation $F + x$ is conducted with a shortcut connection and element-wise addition. Afterward, a second non-linear (i.e., $\sigma(y)$ transformation is applied.

The shortcut connections in Equation (2.1) neither adds an extra parameter nor increases computation complexity and enables a comparisons between plain and residual networks that concurrently have the same number of parameters, depth, width, and computational cost (except for the negligible element-wise addition). The dimensions of x and F in Equation (2.1) must be equal. Alternatively, to match the dimensions, linear projection W_s by the shortcut connections can be applied:

$$y = F(x, \{W_i\}) + W_s x. \tag{2.2}$$

The square matrix W_s can be used in Equation (2.2). However, experiments showed that identity mapping is enough to solve the degradation problem. Therefore, W_s only aims to match the dimensions. Although more levels are possible, it was experimented with function F having two or three layers without stating the exact form of it. Assuming F only has one layer (Equation (2.1)) it is comparable to a linear layer: $y = W_1x + x$. The theoretical notations are about fully-connected layers, but convolutional layers were used. The function $F(x, \{W_i\})$ can be applied to represent multiple convolutional layers. Two feature maps are added element-wise, channel by channel.

2.2.4.2 Network Architectures

Various plain/residual networks were tested to construct an efficient residual network. They trained the network on benchmarked datasets, e.g. the ImageNet dataset, that are used for a comparison of network architectures. Figure (2.2) shows that every residual network needs a plain baseline network inspired by the VGG (Simonyan and Zisserman, 2014) network on which identity mapping by shortcuts is applied.

Plain Network: The philosophy of VGG nets 41 mainly inspires the plain baselines. Two rules convolution layers, which usually have 3×3 filters, follow are:

- feature maps with the same output size have the same number of layers;
- reducing the size of a feature map by half doubles the number of filters per layer to maintain time complexity per layer.

Convolutional layers with a stride of 2 perform downsampling directly. A global average pooling layer and a 1000-way fully-connected layer with softmax are at the end of the network. The number of weighted layers sums up to 34 (Figure 2.25, middle). Compared to VGG nets, this model has fewer filters and lower complexity (Figure 2.25, left).

Residual Network: Based on the above plain network, additional shortcut connections (Figure 2.25, right) turn the network into its associate residual variant. The identity shortcuts (Equation (2.1)) can be directly used in the case of the exact dimensions of the input and output (solid line shortcuts in Figure 2.25). For the different dimensions (dotted line shortcuts in Figure 2.25), two options are considered:

- The shortcut still performs identity mapping, but with extra zero entries padded to cope with the increasing dimensions, without adding new parameters;
- The projection shortcut in Equation (2.2) matches dimensions (due to 1×1 convolutions).

In both cases, shortcuts will be done with a stride of two when they go across feature maps of two sizes.

2.2.5 EfficientNet

Until Tan and Le (2019b) introduced EfficientNet, it was popular to scale only one of the three dimensions – depth, width, or image size. The empirical study shows that it is critical to balance all network dimensions, which can be achieved by simply scaling each with a constant ratio. Based on this observation, a simple yet effective compound scaling method was proposed, which uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients. For example, if $2N$ times more computational resources are available, increasing the network depth by αN, width by βN, and image size by γN would be possible. Here α, β, γ are constant coefficients determined by a small grid search on the original miniature model. Figure 2.26 illustrates the difference between this scaling method and conventional methods. A compound scaling method makes sense if an input image is bigger because a larger receptive field requires more layers and more significant channel features to capture fine-grained patterns. Theoretically and empirically, there has been a special relationship between

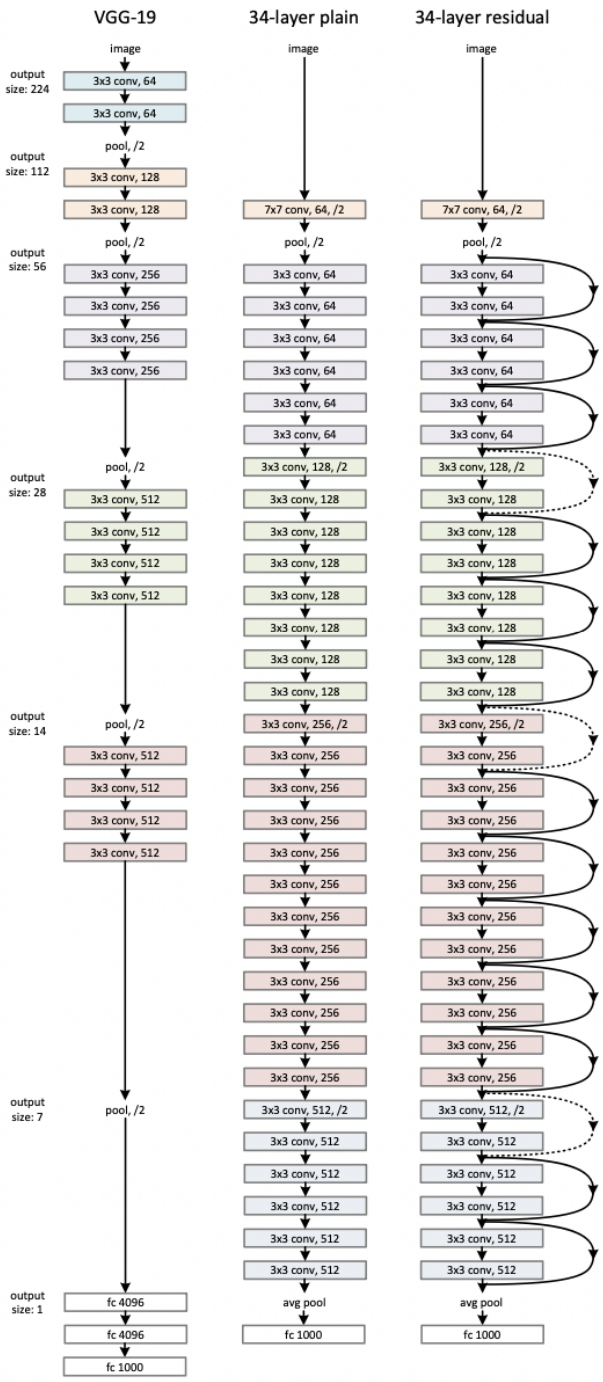


FIGURE 2.25: Architecture of ResNet (He et al., 2015).

network width and depth (Raghu et al., 2016). Existing MobileNets (Howard et al., 2017) and ResNets are used to demonstrated new scaling methods.

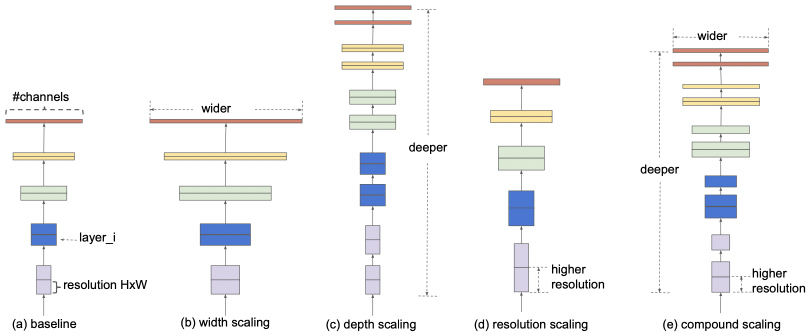


FIGURE 2.26: Model scaling (Tan and Le, 2019b).

2.2.5.1 Compound Model Scaling

2.2.5.1.1 Problem Formulation

A function $Y_i = \mathcal{F}_i(X_i)$ with the operator $\mathcal{F}_i$, output tensor Y_i, input tensor X_i of shape (H_i, W_i, C_i), spatial dimensions H_i, W_i, and channel dimension C_i is called a ConvNet Layer i. A ConvNet N appears as a list of composing layers:

$$\mathcal{N} = \mathcal{F}_{\parallel} \odot \cdots \mathcal{F}_{\in} \odot \mathcal{F}_{\infty}(X_1) = \bigodot j = 1 \cdots k \mathcal{F}_{|}(X_1)$$

Effectively, these layers are often partitioned into multiple stages and all layers in each stage share the same architecture. For example, ResNet has five stages with all layers in every stage being the same convolutional type except for the first layer that performs down-sampling. Therefore, a ConvNet can be defined as:

$$\mathcal{N} = \bigodot_{i=1 \cdots s} \mathcal{F}_{\rangle}^{L_i}\left(X_{(H_i, W_i, C_i)}\right)$$

where $\mathcal{F}_{\rangle}^{L_i}$ denotes layer $\mathcal{F}_{\rangle}$ which is repeated L_i times in stage i, and (H_i, W_i, C_i) is the shape of input tensor X of layer i.

In comparison to the regular ConvNet focusing on the best layer architecture search $\mathcal{F}_{\rangle}$, model scaling centers on the expansion of the network length (L_i), width (C_i), and/or resolution (H_i, W_i) without changing $\mathcal{F}_{\rangle}$ that was predefined in the baseline network. Although model scaling simplifies the design problem of the new resource constraints through fixing $\mathcal{F}_{\rangle}$, a different large design space (L_i, H_i, W_i, C_i) for each layer remains to be explored. To further reduce the design space, all layers are restricted to be scaled uniformly with a constant

ratio. In this case, the goal is to maximize the model's accuracy for any given resource constraint, which is presented as an optimization problem:

$$\begin{aligned} &\max_{d,w,r} \text{Accuracy}\left(\mathcal{N}\left(d,w,r\right)\right) \\ s.t. \mathcal{N}\left(d,w,r\right) = &\bigodot_{I=1\ldots s} \hat{\mathcal{F}}i^{d\cdot\hat{L}i}\left(X_{\left\langle r\cdot\hat{H}_i, r\cdot\hat{W}_i, w\cdot\hat{C}_i\right\rangle}\right) \\ Memory\left(\mathcal{N}\right) \leq &\ targetMemory \\ FLOPS\left(\mathcal{N}\right) \leq &\ targetFlops \end{aligned}$$

where w, d, r are coefficients for scaling network width, depth, and resolution; $\left(\widehat{\mathcal{F}}_i, \widehat{L}_i, \widehat{H}_i, \widehat{W}_i, \widehat{C}_i\right)$ are predefined parameters of the baseline network.

2.2.5.1.2 Scaling Dimensions

The main difficulty of this optimization problem is that the optimal d, w, r depend on each other and the values are changing under different resource constraints. Due to this difficulty, conventional methods mostly scale ConvNets in one of these dimensions:

Depth (d): One of the most significant networks previously described is the ResNet. As it was described, the problem of ResNets is that accuracy gain of a very deep network diminishes. For example, ResNet-1000 has similar accuracy to ResNet-101 even though it contains many more layers.

Width (w): Scaling network width is commonly used for small-sized models. However, wide but shallow networks tend to have difficulty grasping higher-level features.

Resolution (r): Starting from 224×224 in early ConvNets, modern ConvNets tend to use 299×299 or 331×331 for better accuracy. GPipe (Huang et al., 2018) recently achieved state-of-the-art ImageNet accuracy with 480×480 resolution. Higher resolutions, such as 600×600, are also widely used in ConvNets for object detection.

The above analyses lead to the first observation:

Observation 1: Scaling up any network width, depth, or resolution dimension improves accuracy. Without the upscaling, the gain diminishes for bigger models.

2.2.5.1.3 Compound Scaling

Firstly, it was observed that different scaling dimensions are not independent because higher resolution images also require to increase the network depth. The larger receptive fields can help capture similar features that include more pixels in bigger images. Similarly, network width should be increased when the resolution is higher to capture more fine-grained patterns. The intuition

suggests that different scaling dimensions should be coordinated and balanced rather than conventional scaling in single dimensions. To confirm this thought, results of networks with width w without changing depth (d=1.0) and resolution (r=1.0) were compared with deeper (d=2.0) and higher resolution (r=2.0) networks. This showed that width scaling achieves much better accuracy under the same FLOPS. These results lead to the second observation:

Observation 2: To achieve better accuracy and efficiency, balancing the network width, depth, and resolution dimensions during ConvNet scaling is critical. Earlier researches have tried to arbitrarily balance network width and depth, but they all require tedious manual tuning.

A new **compound scaling method**, which uses a compound coefficient φ to uniformly scale network width, depth, and resolution in a principled way was proposed:

$$\begin{aligned} \text{depth:} & \lceil = \alpha^{\varphi} \\ \text{width:} & \sqsupseteq = \beta^{\varphi} \\ \text{resolution:} & \nabla = \gamma^{\varphi} \\ & s.t. \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2 \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1 \end{aligned} \tag{2.3}$$

where α, β, γ are constants that can be determined by a small grid search, φ is a user-specified coefficient that controls how many more resources are available for model scaling, while α, β, γ specify how to assign these extra resources to the network width, depth, and resolution, respectively. Notably, the FLOPS of a regular convolution operation is proportional to d, w^2, r^2, i.e., doubling network depth will double the FLOPS, but doubling network width or resolution will increase the FLOPS by four times. Scaling a ConvNet following Equation (2.3) will approximately increase the total number of FLOPS by $\left(\alpha \cdot \beta^2 \cdot \gamma^2\right)^{\varphi}$. In this chapter, $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$ is constrained such that for any new φ the total number of FLOPS will approximately increase by 2φ.

2.2.5.2 EfficientNet Architecture

A good baseline network is essential because model scaling does not affect its layer operators $F * [i]$. Therefore this method is also estimated on ConvNets. A new mobile-sized baseline called EfficientNet was developed to show the effectiveness of the new scaling method. Metrics that were used to estimate the efficacy are accuracy and FLOPS. The baseline efficient network that was created is named EfficientNet-B0. Afterwards, this compound scaling method is applied in two steps:

- **STEP 1**: By fixing $\varphi = 1$ and, assuming twice more resources available, a small grid search of α, β, γ based on Equation (2.3) showed that the best

values for EfficientNet-B0 are $\alpha = 1.2, \beta = 1.1, \gamma = 1.15$ under the constraint of $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$.

- **STEP 2**: Afterwards, fix α, β, γ as constants and scale up the baseline network with different φ using Equation (2.3) to construct EfficientNet-B1 to B7.

Name	Number of parameters
EfficientNet-B0	5.3M parameters
EfficientNet-B1	7.8M parameters
EfficientNet-B2	9.2M parameters
EfficientNet-B3	12M parameters
EfficientNet-B4	19M parameters
EfficientNet-B5	30M parameters
EfficientNet-B6	43M parameters
EfficientNet-B7	66M parameters

Indeed, even better performance is achievable by searching for α, β, γ directly around a large model, but the search cost becomes prohibitively more expensive on larger models. This method searches once on a small baseline network, then scales the coefficient for all other models.

2.2.5.3 Results and comparison of the networks

To demonstrate the performance of both networks, ResNet and EfficientNets were trained and evaluated on the ImageNet 2012 classification dataset consisting out of 1000 classes. Since deeper scaling should provide better results in the case of ResNet, it was trained with increased depth each time. First meaningful results were obtained in ResNet-34, which performed 3.5 % better than plain-34 baseline when top-1 accuracy is compared. They also compared three versions of ResNet: (A) zero-padding shortcuts (increasing dimensions, all shortcuts are parameter-free) (B) projection shortcuts (increasing dimensions, other shortcuts are identity), and (C) all shortcuts are projections. Each version improved both, the top-1 and top-5 accuracy. Afterward, the depth of the network was increased and ResNet-50, ResNet-101, and ResNet-152 were created. Each increase in depth leads to higher accuracy. In deeper models, the trade-off between accuracy increase and deeper model is not worth describing. All results are shown in the following table:

Model	top-1 acc.	top-5 acc.
VGG-16	71.93	90.67
GoogLeNet	-	90.85
plain-34	71.46	89.98
ResNet-34 A	74.97	92.24

Model	top-1 acc.	top-5 acc.
ResNet-34 B	75.48	92.54
ResNet-34 C	75.81	92.6
ResNet-50	77.15	93.29
ResNet-101	78.25	93.95
ResNet-152	**78.57**	**94.29**

In the case of EfficientNets, the results achieved by the previous state-of-the-art networks on the same ImageNet dataset were aimed to improve. Among all state-of-the-art networks, EfficientNets were compared with ResNets-50 and ResNet-152. They compared the results of networks deviated by changing scaling parameters EfficientNet-B0 to EfficientNet-B7. The results of each network were better than the previous one. Also, they have shown that EfficientNet-B0 outperforms ResNet-50 and that EfficientNet-B1 outperforms ResNet-152. This means that scaling through all three dimensions can provide better results than scaling through just one dimension. The drawback of this approach is the computational power which makes it less popular than the previous methods. Again, all results are shown in the following table:

Model	top-1 acc.	top-5 acc.
EfficientNet-B0 / ResNet-50	77.1 / 76	93.3 / 93
EfficientNet-B1 / ResNet-152	79.1 / 77.8	94.4 / 93.8
EfficientNet-B2	80.1	94.9
EfficientNet-B3 / ResNeXt-101	81.6 / 80.9	95.7 / 95.6
EfficientNet-B4	82.9	96.4
EfficientNet-B5	83.6	96.7
EfficientNet-B6	84	96.8
EfficientNet-B7 / GPipe	**84.3** / 84.3	**97** / 97

2.2.6 Contrastive learning

In recent years the problem of classification of unlabeled dataset is becoming more widespread. More unlabeled datasets requiring human labeling are created in fields like medicine, the automotive industry, military, etc. Since the process is expensive and time-consuming, researchers assumed it could be automated with contrastive learning frameworks. One of the first and most known contrastive learning frameworks is SimCLR (Chen et al., 2020a). The advantage of this framework is its simplicity, yet it achieves high accuracy on classification tasks. The main idea is to have two copies of the image, which are then used to train two networks and that are compared. The problem with this framework is that it doubles the size of the dataset and reaches among all images, which can be computationally infeasible for large datasets. Bootstrap Your Own Latent

(Grill et al., 2020b) was introduced to avoid making double-sized datasets. The idea was to bootstrap image representations to avoid unnecessary image comparisons. These two frameworks will be described in this chapter.

Further improvements in the choice of creating two views of images and comparison techniques were presented in different frameworks such as Nearest-Neighbor Contrastive Learning (NNCLR) (Dwibedi et al., 2021), Open World Object Detection (ORE) (Joseph et al., 2021), Swapping Assignments between multiple Views (SwAV) {Caron et al. (2020)}, and many more. This field is a constant research topic and new improved frameworks are proposed on a constant basis to help researchers solve different tasks that requires labeled datasets.

2.2.6.1 A Simple Framework for Contrastive Learning of Visual Representations

Chen et al. (2020a) intended to analyze and describe a better approach to learning visual representations without human supervision. They have introduced a simple framework for contrastive learning of visual representations called SimCLR. As they claim, SimCLR outperforms previous work, is more straightforward, and does not require a memory bank.

Intending to understand what qualifies good contrastive representation learning, the significant components of the framework were studied and resulted in:

- A contrastive prediction task requires combining multiple data augmentation operations, which results in effective representations. Unsupervised contrastive learning benefits from more significant data augmentation.
- The quality of the learned representations can be substantially improved by introducing a learn-able non-linear transformation between the representation and the contrastive loss.
- Representation learning with contrastive cross-entropy loss can be improved by normalizing embeddings and adjusting the temperature parameter appropriately.
- Unlike its supervised counterpart, contrastive learning benefits from larger batch sizes and extended training periods. Contrastive learning also benefits from deeper and broader networks, just as supervised learning does.

2.2.6.2 The Contrastive Learning Framework

Like for SimCLR, a contrastive loss is used to learn a representation by maximizing the agreement between various augmented views of the same data example. This framework contains four significant components, which are shown in Figure 2.27:

1. A stochastic *data augmentation* module
2. A neural network *base encoder*

3. A small neural network *projection head*
4. A *contrastive loss function*

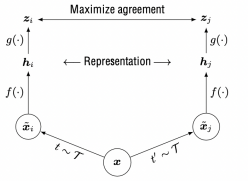


FIGURE 2.27: A simple framework for contrastive learning of visual representations (Chen et al., 2020a).

2.2.6.2.1 Stochastic data augmentation module

First, the minibatch of N examples is sampled randomly, and the contrastive prediction task is defined on pairs of augmented examples, resulting in $2N$ data points. A memory bank was not used to train the model, instead, the training batch size varies from 256 to 8192. Any given data example randomly returns two correlated views of the same example, denoted $\tilde{x}_i$ and $\tilde{x}_j$, which is known as a **positive pair**. **Negative pairs** are all other $2(N-1)$ pairs. In one view, some data augmentation techniques are applied. Data augmentation is widely embraced in supervised and unsupervised representation learning. Unfortunately, it has not been used to define the contrastive prediction task, which is mainly determined by changing the architecture. It was shown that choosing different data augmentation techniques can reduce the complexity of previous contrastive learning frameworks. There are many data augmentation operations, the focus was on the most common ones, which are:

- **Spatial geometric transformation**: cropping and resizing (with horizontal flipping), rotation and cutout,
- **Appearance transformation**: color distortion (including color dropping), brightness, contrast, saturation, Gaussian blur, and Sobel filtering.

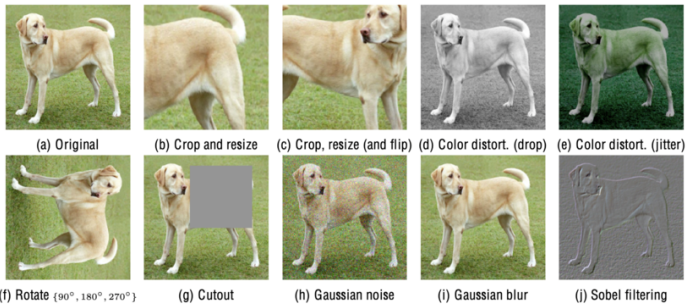


FIGURE 2.28: Augmentation texhniques (Chen et al., 2020a).

Due to the image sizes in the ImageNet dataset, all images were always randomly cropped and resized to the same resolution. Later on, other targeted data augmentation transformations were applied to one branch, remaining the one as original i.e. $t(x_i) = x_i$. Applying just individual transformation is insufficient for the model to learn good representations. The model's performance improves after composing augmentations, although the contrastive prediction task becomes more complex. The composition of augmentations that stood out were random cropping and random color distortion.

It was also observed that stronger color augmentation significantly improves the linear evaluation of unsupervised learned models. Stronger color augmentations do not enhance the performance of supervised learning models when trained with the same augmentations. Based on the experiments, unsupervised contrastive learning benefits from stronger color data augmentation than supervised learning.

2.2.6.2.2 Neural network base encoder

Neural network based encoder $f(\cdot)$ extracts multiple representation vectors from the augmented data examples. This framework does not restrict a choice of the network architecture, although for simplicity, the commonly used ResNet was picked and gives $h_i = f(\tilde{x}_i) = ResNet(\tilde{x}_i)$ where $\mathbf{h}_i \in \mathbb{R}^d$ is the output after the average pooling layer. Although increasing depth and width improves performance, the ResNet-50 was chosen. Furthermore, when the model size increases, the gap between supervised and unsupervised learning shrinks, suggesting that bigger models benefit more from unsupervised learning.

2.2.6.2.3 Small neural network projection head

A small neural network projection head $g(\cdot)$ maps the representation to the space where the contrastive loss is applied to. The importance of including a projection head, i.e., $g(h)$ was evaluated and they considered three different architectures for the head:

1. identity mapping,
2. linear projection,
3. the default non-linear projection with one additional hidden layer and ReLU activation function.

The results showed that a non-linear projection head is better than a linear projection and much better than no projection. It improves the representation quality of the layer that is applied previous to it. They have used a MLP with one hidden layer to obtain $z_i = g(\mathbf{h}_i) = W^{(2)}\sigma\left(W^{(1)}\mathbf{h}_i\right)$ where σ is a ReLU non-linearity transformation.

This step is performed because defining the contrastive loss on z_i instead of on $\mathbf{h}_i$ would not lead to a loss of information caused by contrastive loss. Especially,

$z = g(h)$ is trained to be invariant to data transformations. As a result, g can remove information useful for a downstream task such as object color or orientation. Using the non-linear transformation $g(*)$, h can maintain and form more information.

2.2.6.2.4 Contrastive loss function

Given a set $\{\tilde{x}_{ik}\}$ including a positive pair of examples $\tilde{x}_i$ and $\tilde{x}_j$, the contrastive prediction task aims to identify $\tilde{x}_i$ in $\{\tilde{x}_i\}_{k \neq i}$ for a given $\tilde{x}_i$. In the case of positive examples, the loss function is defined as

$$\updownarrow_{i,j} = -\log \frac{\exp\left(\frac{\text{sim}(z_i, z_j)}{\tau}\right)}{\sum_{k=1}^{2N} \mathbb{I}_{[k \neq i]} \exp\left(\frac{\text{sim}(z_i, z_k)}{\tau}\right)}$$

where $\mathbb{I}_{[k \neq i]} \in \{0, 1\}$ is an indicator function, τ denotes a temperature parameter and $\text{sim}(\mathbf{u},\mathbf{v}) = \frac{\mathbf{u}^T\mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}$ is a dot product between $\updownarrow_2$ and normalized $\mathbf{u}, \mathbf{v}$.

The final loss is calculated across all positive pairs, both (i, j) and (j, i), in a mini-batch. It was named **NT-Xent**, the normalized temperature-scaled cross-entropy loss.

The NT-Xent loss was compared against other commonly used contrastive loss functions, such as logistic loss and margin loss. Gradient analysis shows that l_2 normalization, cosine similarity, and temperature together effectively weight different examples and a suitable temperature can make the model learn from hard negatives. The advantage of NT-Xent is that it weights the negatives by their relative hardness. Without normalization and proper temperature scaling the performance is significantly worse. Also, the contrastive task accuracy is higher, but the resulting representation is worse under linear evaluation.

2.2.6.3 Bootstrap Your Own Latent

The fundamental idea of contrastive learning is to create pairs of images on which the framework would be trained. Creating negative pairs relies on large batch sizes, memory banks, or customized mining strategies which can be challenging in larger datasets. Grill et al. (2020b) wanted to create a new approach that would achieve better performance than other contrastive methods without using negative pairs. A solution they have introduced is a method called Bootstrap Your Own Latent (BYOL). The idea was to bootstrap representations of images. As a result, BYOL is more robust to the choice of image augmentations. Furthermore, BYOL has two neural networks, called online and target network, who interact and learn from each other. Using an augmented view of an image, BYOL trains its online network to predict the target network's representation of another augmented view. This approach achieved state-of-the-art results when trained on the ImageNet dataset under

the linear evaluation protocol. Additionally, compared to SimCLR, a strong contrastive baseline, BYOL suffers from much less performance drop when only random crops are used to augment images.

2.2.6.3.1 Description of the method

BYOL aims to learn a representation of y_θ. It uses two neural networks: *online* and *the target network* to achieve that. The *online network* is determined by a set of weights θ and consists of:

- an encoder f_θ,
- a projector g_θ,
- a predictor q_θ.

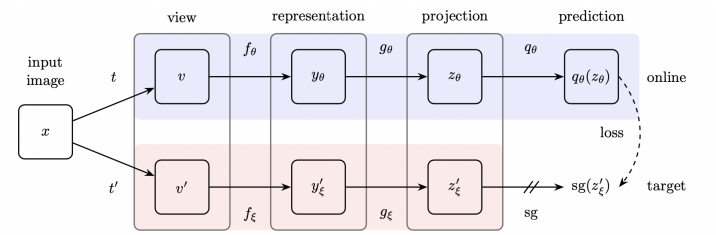


FIGURE 2.29: Bootstrap Your Own Latent (Grill et al., 2020b).

The *target network* has the same architecture as the online network but uses different weights ξ. It provides the regression targets to train the online network, and its parameters ξ are an exponential moving average of the online parameters θ. Precisely, given a target decay rate $\tau \in [0, 1]$, after each training step, the following update

$$\xi \leftarrow \tau\xi + (1 - \tau)\theta$$

is performed. Firstly, an image is sampled uniformly from $\mathcal{D}$ from which two distributions of image augmentations $\mathcal{T}$ and $\mathcal{T}'$ are created. BYOL applies respectively two image augmentations $t \sim \mathcal{T}$ and $t' \sim \mathcal{T}'$ creating two augmented views $v \triangleq t(x)$ and $v' \triangleq t'(x)$. First augmented view v is used for the online network and result in the output $y_\theta \triangleq f_\theta(v)$ and afterwards the projection $z_\theta \triangleq g_\theta(y)$. Similarly, from the second augmented view v' the target network outputs $y'_\xi \triangleq f_\xi(v')$ and the target projection $z'_\xi \triangleq g_\xi(y')$. Later on output a prediction of $q_\theta(z_\theta)$ of z'_ξ and ℓ_2-normalize both $q_\theta(z_\theta)$ and z'_ξ to

$$\overline{q_\theta}(z_\theta) \triangleq q_\theta(z_\theta) / \|q_\theta(z_\theta)\|_2 \quad \text{and} \quad \bar{z}'_\xi \triangleq z'_\xi / \left\|z'_\xi\right\|_2 .$$

The predictor is only applied to the online pipeline, making the architecture asymmetric between the online and target pipeline. Lastly, the following mean

squared error between the normalized predictions and target projections is defined:

$$\mathcal{L}_{\theta,\xi} \triangleq \left\|\overline{q_\theta}\left(z_\theta\right) - \bar{z}'_\xi\right\|_2^2 = 2 - 2 \cdot \frac{\left\langle q_\theta\left(z_\theta\right), z'_\xi \right\rangle}{\left\|q_\theta\left(z_\theta\right)\right\|_2 \cdot \left\|z'_\xi\right\|_2}$$

The loss is symmetrized $\mathcal{L}_{\theta,\xi}$ by using v' for the online network and v for the target network separately to calculate $\widetilde{\mathcal{L}}_{\theta,\xi}$. At each training step, a stochastic optimization step is applied to minimize $\mathcal{L}_{\theta,\xi}^{\mathrm{BYOL}} = \mathcal{L}_{\theta,\xi} + \widetilde{\mathcal{L}}_{\theta,\xi}$ with respect to θ only but not ξ. BYOL's dynamics are summarized as

$$\theta \leftarrow \text{optimizer}\left(\theta, \nabla_\theta \mathcal{L}_{\theta,\xi}^{\mathrm{BYOL}}, \eta\right).$$

where η is a learning rate. At the end of the training, only the encoder f_θ is used.

2.2.6.4 Comparison of contrastive learning frameworks

Of all frameworks, SimCLR is the most popular due to its simplicity. The ResNet-50 in 3 different hidden layer widths (width multipliers of 1×, 2×, and 4×) were used and trained for 1000 epochs each. The accuracy of these frameworks on the ImageNet dataset with few labels improved when the width of ResNet-50 increases. For SimCLR with ResNet-50 top-1 accuracy is 69.3 and top-5 accuracy is 89, while for ResNet-50(4x) top-1 accuracy is 85.8 and top-5 accuracy is 92.6. These results are comparable with supervised methods. The BYOL framework was built to improve the results of SimCLR. It was also stated that the accuracy for the baseline ResNet-50 is 74.3 and 91.6 for top-1 accuracy and top-5 accuracy. When using ResNet-50(4x), an increase in accuracy to 78.6 and 94.2 for top-1 and top-5 is observed, respectively. More information about performance can be found in following table:

Model	Architecture	Param (M)	top-1 acc.	top-5 acc.
SimCLR	ResNet-50	24	69.3	89.0
SimCLR	ResNet-50 (2x)	94	74.2	93.0
SimCLR	ResNet-50 (4x)	375	76.5	93.2
BYOL	ResNet-50	24	74.3	91.6
BYOL	ResNet-50 (x2)	94	77.4	93.6
BYOL	ResNet-50 (x4)	375	78.6	94.2
BYOL	ResNet-200 (x2)	250	79.6	94.8

2.2.7 Transformers in Computer Vision

Since the first appearance of the Transformers architecture in 2017 **?**, it has become an irreplaceable part of all-natural language processing (NLP) models. The main advantage of Transformers is that they can be trained on a large text corpus and then fine-tuned on a smaller task-specific dataset. This enabled model training of unspecified size with more than 100B parameters.

However, computer vision still relied on convolutional architectures. With datasets constantly growing and the diversity of the fields computer vision tasks could be applied to, researchers wanted to implement Transformers architecture in the CV field. Some works aim for combining CNN-like architectures with self-attention (Wang and Li, 2018). Others attempted to replace convolutions entirely, e.g. Ramachandran et al. (2019). Due to specialized attention patterns, the problem was that they have not yet been scaled effectively on modern hardware accelerators. Therefore, in large-scale image recognition, classic ResNet-like architectures are still state-of-the-art.

In 2021 the Google research Brain Team published the paper "An image is worth 16×16 words" where they introduced new Transformers-based architecture for CV called Vision Transformers (ViT) (Dosovitskiy et al., 2020c). Based on the success of Transformer in NLP scaling, they aimed to apply standard Transformer directly to images with little as possible changes to the existing architecture. The image is split into patches and linear embeddings of these patches are provided as inputs to the Transformer. These patches are the same as tokens (e.g. words) in NLP. The model is trained for image classification in a supervised learning fashion.

2.2.7.1 Vision Transformers

Brain Team wanted to create simple but universally scalable architecture to follow the original Transformers architecture.

2.2.7.1.1 Method

Compared to NLP, with 1-dimensional token embedding input for the Transformer, images are 2-dimensional objects. Firstly, images needed to be represented differently to imitate original architectures as close as possible. For that reason image $x \in \mathbb{R}^{H \times W \times C}$ is reshaped into a sequence of flattened 2-dimensional patches $x_p \in \mathbb{R}^{N \times \left(P^2 \cdot C\right)}$, where (H, W) is the resolution of the original image, C is the number of channels, (P, P) is the resolution of each image patch, and $N = HW/P^2$ is the resulting number of patches, also the Transformer's effective input sequence length. The Transformer input through all layers is a fixed vector of size D. The first step is to flatten the patches, usually 16×16 and map them to D dimensions with a trainable linear projection to create patch embeddings.

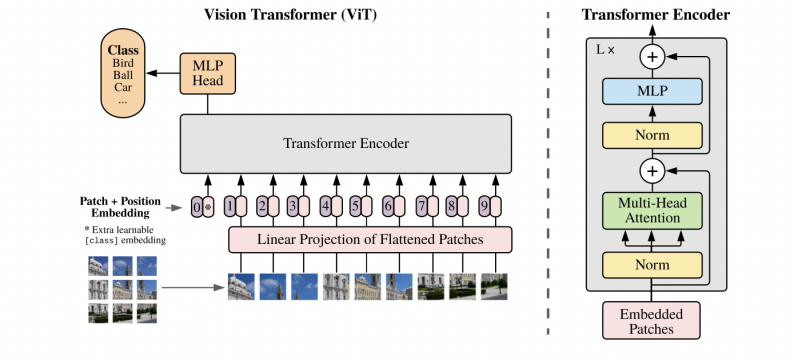


FIGURE 2.30: Vision Transformer (Dosovitskiy et al., 2020c).

$$\mathbf{z}_0 = \left[\mathbf{x}_{\text{class}}\,; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}\right] + \mathbf{E}_{pos}, \mathbf{E} \in \mathbb{R}^{\left(P^2 \cdot C\right) \times D}, \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$$

To this sequence of "patch embeddings", a prefix learnable [class] token, like in BERT, is usually added. This token $\mathbf{z}_0^0 = \mathbf{x}_{class}$ tells the model to classify the image and increases the dimension of vector z. Also, the state of this token at the output of the Transformer encoder $\left(\mathbf{z}_L^0\right)$, on which the layernorm is applied, serves as the image representation y.

$$\mathbf{y} = \text{LN}\left(\mathbf{z}_L^0\right)$$

Furthermore, it is the only one to which the classification head is attached to during pre-training and fine-tuning. The classification head during pre-training is compiled of MLP with one hidden layer and a single linear layer at a fine-tuning time. Position embedding, a standard learnable 1-dimensional position embedding, are attached to the patch embeddings, serving as input to the encoder. The standard Transformer encoder consists of alternating layers of multiheaded self-attention and MLP blocks. After each block, a residual connection is applied.

$$\mathbf{z}'_\ell = \text{MSA}\left(\text{LN}\left(\mathbf{z}_{\ell-1}\right)\right) + \mathbf{z}_{\ell-1}, \ell = 1 \ldots L$$

$$\mathbf{z}_\ell = \text{MLP}\left(\text{LN}\left(\mathbf{z}'_\ell\right)\right) + \mathbf{z}'_\ell, \ell = 1 \ldots L$$

Vision Transformer has a significantly lower inductive bias than CNNs in image-specific information. VIT only has local and translational equivariant MLP layers, while the self-attention layers are global. A 2-dimensional neighborhood structure is used sparingly: the image is cut into patches at the beginning

and the position embeddings are resized as needed at the fine-tuning time. Alternatively, the input sequence can consist of a CNN's feature map on which the patch embedding projection is applied. Vision Transformers are pre-trained on large datasets and fine-tuned to (smaller) downstream tasks. For fine-tuning, a projection head is removed and a zero-initialized $D \times K$ feedforward layer is attached with K being the number of downstream classes. It is also beneficial to use higher resolution then in pre-training. Also ViT can handle arbitrary sequence lengths but the pre-trained position embeddings can become sufficient. It is necessary to point out that resolution adjustment and patch extraction are the only points at which an inductive bias about the 2-dimensional structure of the images is manually injected into the Vision Transformers

2.2.7.1.2 Experiments

Similarly to BERT models, multiple versions of the model at various scales were created. They have created Base = "B", Large = "L", Huge = "H" versions of ViT, with 12, 24 and 32 layers and 86M, 307M and 632M parameters respectively.

To explore the model scalability, the previous mentioned dataset ImageNet was used. In addition, ViT was compared against a slightly modified ResNet called "ResNet(BiT)". The batch Normalization layer was replaced with Group Normalization and used standardized convolutions. Another network that it was compared to was Noisy Student (Xie et al., 2019), a large EfficientNet. Experiments showed that ViT Hughe with 14×14 input patch size outperformed both CNN-based networks with an accuracy of 88.5%, whereas ResNet BiT had 87.54% and Noisy Student 88.4%. It is worth mentioning that ViT Large with 16×16 input patch size had 87.76% accuracy on the same dataset. Another thing worth pointing out is that ViT outperforms CNN-based architectures on all larger datasets yet performs slightly worse than CNN networks on a smaller dataset.

2.2.8 Conclusion

In this chapter, the authors presented some of the current state-of-the-art approaches in Computer Vision. Nowadays, when technology is advancing each day, creating networks that would imitate human brain is more challenging. Still, the networks presented in this chapter are highly accurate and creating network which can out-perform them is challenging. Furthermore, it is noticeable that the application of CV is dictating the development of networks and frameworks which help humans with their everyday tasks.

2.3 Resources and Benchmarks for NLP, CV and multimodal tasks

Author: Christopher Marquardt

Supervisor: Christian Heumann

When we see athletes perform in their sports we only see the results of their hard work prior or till to the event. Most of the time they casually talk about their off-season, but everybody knows the results are made in the off-season.

Same goes for the models we will see in the later chapters. We are just interested in the results, but why and how does the model come to these results? It has to learn to some key fundamentals of the modality to achieve these results. But how do they get them to perform in such a way or even better? It's possible to build better architectures and/or use more and new data to achieve this. New data by hand is easy to get but this new data results in a new problem. New data has to be carefully labeled by humans, which can be very expensive by the amount of data. Models which learn from labeled data use the supervised learning strategy. This learning strategy is a bottleneck for future progress, because of the given reason.

But the need for labeling the data isn't the only problem. Let's visit the athlete analogy again. Imagine a professional football player has to participate in a professional ski race. He will not be able to compete with the others, because they are trained only to do ski races. Here see the other problem. Models which use supervised learning have shown to perform very well on the task they are trained to do. This means models which learn on carefully labeled data only perform very well on this specific task, but poor on others. Also it's not possible to label everything in the world.

So the goal is to generate more generalist models which can perform well on different tasks without the need of huge labeled data. Humans are able to perform well on different tasks in a short amount of time. Humans, for example, only need a small amount of hours to learn how to drive a car, even without supervision. On the other hand fully automated driving AI need thousand of hours of data to drive a car. Why do humans learn so fast compared to machines? Humans don't rely on labeled data, because most of the time humans learn by observation. By this humans generate a basic knowledge of how the world works, which also called common sense. This enables us to learn so much faster compared to machines. Meta AI (Yann and Ishan, 2021) believes that self-supervised learning is one of the most promising ways to generate background knowledge and some sort of common sense in AI systems. By self-supervised learning one means a supervised learning algorithm, but it doesn't need an external supervisor. Self-supervised pre-training differs

between the modalities, which means there is not an approach which works in all modalities. The following chapter will inspect on the one hand pre-training resources and the use of them and on the other hand also the benchmarks which are used for Natural Language Processing (NLP), Computer Vision (CV) and ,the combination of both, vision language pre-trained models (VL-PTM).

2.3.1 Datasets

After pointing out that pre-training is very important, one might ask how do the datasets look and how do the different modalities pre-train? At first we will inspect the former one and focus afterwards on the use of the resources. As one might expect NLP models pre-train on text, CV models pre-train on images and VL-PTM pre-train on text image pairs, which can somehow be seen as a combination of NLP and CV. But CV models mostly used labeled data like a picture of a dog with the corresponding single label "dog". MML datasets can contain several sentences of text which correspond to the given image.

Even if the datasets might be completely different, the procedure to get the data is mostly the same for all of them, because the data is crafted from the internet. This can lead to a problem, since by using this method the resulting dataset might be noisy. One approach for the VL-PTM, for example, is to use CommonCrawl and extract the image plus the alt of an image. The alt is an alternate text for an image, if the image cannot be displayed or for visual impaired people. This seems like a reasonable approach, but the alt is often not very informative about what's in the image.

Another difference between the modalities is the cardinality of the pre-training data. It's easy to realize that text is by far easiest to crawl from the internet. This results in huge high-quality massive text data. Some magnitudes smaller are the datasets for CV. Since VL-PTM are pretty new compared to the other modalities it still relatively small, but growing fast. A small downer is that some of the datasets are not public available. The big companies like to keep their models and used datasets private, which hinders the reproducibility, but there are also real open AI competitors like LAION and Eleuther in the field. The next chapter will provide some of the most used pre-training datasets.

2.3.1.1 Natural Language Processing Datasets

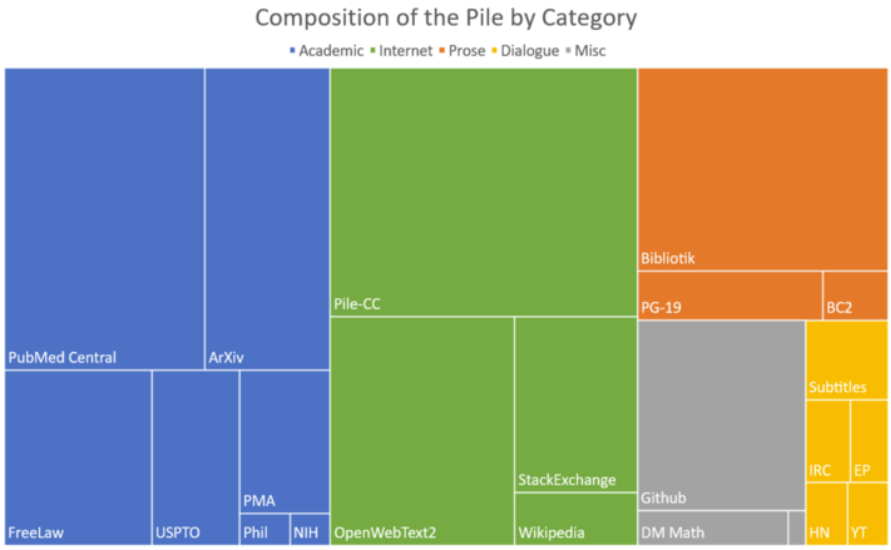
2.3.1.1.1 Common Crawl

As already mentioned, extracting text from the internet is rather easy. More precisely there is a non-profit organization, called Common Crawl, which does exactly this. They provide copies of the internet to researchers, companies and individuals at no cost for the purpose of research and analysis. The Common Crawl corpus contains petabytes of data collected since 2008. Every month, Common Crawl releases a snapshot of the web obtained by randomly exploring

and sampling URLs. It contains raw web page data, extracted metadata and text extractions. The advantages of Common Crawl come along with their disadvantages. The text is from diverse domains but with varying quality of data. To handle the raw nature of the datasets one often has to use a well-designed extraction and filter to use the datasets appropriately (Gao et al., 2020). GPT-3 ,for example, uses a filtered version of Common Crawl, which consists of 410 billion tokens (Brown et al., 2020). So data for NLP is freely available but one needs to use well-designed extraction and filtering to really use the dataset.

2.3.1.1.2 The Pile

Recent work (Rosset, 2020) showed that diversity in training datasets improves general cross-domain knowledge and downstream generalization capability for language models. The Pile (Gao et al., 2020) was introduced to address exactly these results. The Pile contains 22 sub-datasets, including established NLP datasets, but also several newly introduced ones. The size of the 22 sub-datasets, which can be categorized roughly into five categories, pile up to around 825 GB of data. The following treemap shows the distribution of the dataset.



While only 13% of the world's population speaks English, the vast majority of NLP research is done on English. Gao et al. (2020) followed this trend, but did not explicitly filtered out other languages when collecting our the data. This leads to the fact that roughly 95% of the Pile is English. Also EuroParl (Koehn, 2005), a multilingual parallel corpus introduced for machine translation, is included in the Pile. To train GPT-2 Open AI collected data from WebText. WebText is an internet dataset created by scraping URLs extracted from

Reddit submissions with a minimum score for quality, but sadly it was never released to the public. Independent researchers reproduced the pipeline and released the resulting dataset, called OpenWebTextCorpus (Gokaslan and Cohen, 2019) (OWT). Eleuther created an enhanced version of the original OWT Corpus called OpenWebText2. It covers all Reddit submissions from 2005 up until April 2020. It covers content from multiple languages, document metadata, multiple dataset versions, and open source replication code.

They also explicitly included a dataset of mathematical problems (DeepMind Mathematics) to improve the mathematical ability of language models trained on the Pile. An ArXiv dataset was in included in the hopes that it will be a source of high quality text and math knowledge, and benefit potential downstream applications to research in these areas and also because arXiv papers are written in LaTeX. Training a language model to be able to generate papers written in LaTeX could be a huge benefit to the research community.

Since CC needs further steps, due to the raw nature of CC, to really use is. Pile-CC is Common Crawl-based dataset, which can be used directly. It yields higher quality output than directly using the WET files. These were only some of the 22 included datasets. A more detailed description of the sub-dataset and the reasons why these were included can be found in the corresponding paper (Gao et al., 2020).

2.3.1.1.3 Multilingual Datasets

Another pre-cleaned version of CC is CC-100 (Wenzek et al., 2019). They present a pipeline to create curated monolingual corpora in more than 100 languages. A filter, which covers the data based on their distance to Wikipedia, is used and this improves the quality of the resulting dataset. However, its English portion is much smaller than the Pile. But a multilingual dataset might help a low-resource language acquire extra knowledge from other languages. Perhaps the most multilingual corpus publicly available, containing 30k sentences in over 900 languages, is the Bible corpus (Mayer and Cysouw, 2014). Till now all datasets were freely available and almost directly usable. The next one is not public available for some reasons.

To provide mT5 (Xue et al., 2020), which is multilingual pre-trained text-to-text transformer, a suitable pre-training dataset, Google Research designed a dataset including more than 100 languages. The dataset is called mC4 (Xue et al., 2020). Since some languages are relatively scarce on the internet, they used all of the 71 monthly web scrapes released so far by Common Crawl. It contains 6.6 billion pages and 6.3 trillion tokens. A smaller version of the mC4 is also used by Google Research. The smaller dataset C4 (Colossal Clean Common Crawl) was explicitly designed to be English only. The C4 dataset is a collection of about 750GB of English-language text sourced from the public Common Crawl web.

Most of the datasets used in NLP are derived entirely from Common Crawl and Rosset (2020) came to the result, that the current best practice in training large-scale language models involve using both large web scrapes and more targeted, higher-quality datasets, which the Pile directly addresses.

2.3.1.1.4 BooksCorpus

The last dataset for NLP is the BooksCorpus dataset (Zhu et al., 2015). The BooksCorpus uses books from yet unplished authors from the web. Only books with more than 20k words were included to filter out shorter, noisier stories. This results in around 11k books from 16 different genres. So more than 74 million sentences can be used in pre-training. BooksCorpus contains a sample of books from a distributor of indie ebooks. Sadly a datasheet about the BooksCorpus was not releasd with the corresponing paper.

Frankly, there was just an paragraph about the content and the extraction inside the paper (Zhu et al., 2015). Bandy and Vincent (2021) addressed exactly this short coming. They provided a retrospective datasheet about the BooksCorpus. Some of their major concerns were copyright violations, duplicate books, skewed genre representation, potentially skewed religious representation and also problematic content (18+ content). Little harm can be expected if an informed adults reads books with these concers, but how does a language model contribute to for example well-documented gender discrimination if it trains on these books.

Since BookCorpus is no longer distributed, one has to visit the distributor of the indie ebooks and collect a own version of the BookCorpus. This is one of the user-based dataset, besides to the datasets of the Pile.

2.3.1.2 Computer Vision Dataset

2.3.1.2.1 ImageNet

The next inspected modality is CV. Almost every state-of-the-art CV model uses a classifier pre-trained on an ImageNet based dataset. ImageNet uses the hierarchical structure of WordNet (Fellbaum, 2010). At the release of ImageNet-1k the amount of classes was unheard at this time point. Datasets like CIFAR-10 (Krizhevsky et al., 2009) and CIFAR-100 (Krizhevsky et al., 2009) had 10 or 100 classes, but ImageNet1k had 1000 different classes and this was not the only major improvement. They also increased the resolution from 32×32 to 256×256. In all, there are roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images. The ImageNet-1k dataset is a subset of the ImageNet dataset (Deng et al., 2009). The full ImageNet dataset is also called ImageNet-21k. It consists of more than 14 million images, divided in almost 22k classes. Because of this some paper described it as ImageNet-22k.

Those two dataset do not only differ by the amount of classes, but also by the type of labels. The labels of ImageNet-21k are not mutually exclusive.

Because of this the pre-training wiht ImageNet-1k is far more popular. Also the ImageNet-21k dataset lacks an official train-validation split, which is just another reason why ImageNet-1k is more popular. The raw dataset ImageNet-21k is around 1.3 terabyte (TB). It's also nice, that the the dataset of ImageNet are open available. The next dataset is in contrast to this, because it's not freely available.

2.3.1.2.2 Joint-Foto-Tree (JFT) & Entity-Foto-Tree (EFT)

The Joint-Foto-Tree (JFT) 300M is one of the follow up version of the JFT dataset (Hinton et al., 2015b). Given the name it consists of 300 million images and on average each image has 1.26 labels. The whole datasets has around 375 million labels. These labels can be divided into 18291 classes. These categories form a rich hierarchy with the maximum depth of hierarchy being 12 and maximum number of child for parent node being 2876 (Sun et al., 2017). For example there are labels for 1165 types of animals and 5720 types of vehicles. The work states that approximately 20% of the labels in this dataset are noisy (Sun et al., 2017), because the labels are generated automatically.

It also provides the fact, that the distribution is heavily long-tailed, which means that some of the classes have less than 100 images. There is also an extendend version of the JFT dataset.

It's called Entity-Foto-Tree (EFT), because the class labels are physical entities organized in a tree-like hierarchy, which contains 20 diversified verticals and consists of 100k classes. It's even rarely used in practice by Google because of the intolerable large model size and the slow training speed (Gao et al., 2017). Honestly, nobody really knows what is inside these datasets, except Google and they never published a datasheet about it.

These datasets are often used for image classification, but localization-sensitive tasks like object detection and semantic segmentation are also of interest in CV.

2.3.1.2.3 Objects365

Objects365 (Shao et al., 2019) is a large-scale object detection and semantic segmentation freely available dataset. It contains 365 object categories with over 600K training images. More than 10 million, high-quality bounding boxes are manually labeled through a three-step, carefully designed annotation pipeline. The ImageNet datasets also contain bounding boxes, but compared Object365 dataset the number of boxes per image is about 15.8 vs 1.1 (Deng et al., 2009). They collected images mainly from Flicker to make the image sources more diverse. All the images conform to licensing for research purposes. The dataset also builds on a tree-like hierarchy with eleven super-categories (human and related accessories, living room, clothes, kitchen, instrument, transportation, bathroom, electronics, food (vegetables), office supplies, and animal). Further

they proposed 442 categories which widely exists in daily lives. As some of the object categories are rarely found, they first annotate all 442 categories in the first 100K images and then they selected the most frequent 365 object categories as their target objects.

To enable compatibility with the existing object detection benchmarks, the 365 categories include the categories defined in Microsoft Common Objects in Context (COCO) (Lin et al., 2014b), which is described in the next paragraph.

2.3.1.2.4 Microsoft Common Objects in Context (COCO)

Microsoft decided to employed a novel pipeline for gathering data with extensive use of Amazon Mechanical Turk. Their goal was to create a non-iconic image collection. Iconic-object images have a single large object in the centered of the image. By this they provide high quality object instances, but they also lack information of contextual important and non-canonical viewpoints (Lin et al., 2014b). Recent work showed that non-iconic images are better at generalizing (Torralba and Efros, 2011). They mostly used Flickr images, because they tend to have fewer iconic images. This results in a collection of 328,000 images. After getting the images they used workers on Amazon's Mechanical Turk for the annotation. The workers got a list with 91 categories and 11 super-categories. At first a worker had to decide if a super-category (e.g. animal) was present or not. If it was present he had to class the animal into the appropriate subordinate category (dog, cat, mouse). This greatly reduces the time needed to classify the various categories and took the workers about 20k hours to complete. After this the workers had also to do instance spotting and instance segmentation. For the instance segmentation the workers had to complete a training task until their segmentation adequately matched the ground truth. Only 1 in 3 workers passed this training stage. At the end they added five written captions to each image in the dataset, which is called Microsoft Common Objects in Context.

At the end they utilized more than 70,000 worker hours to collect a amount of annotated object instances, which were gathered to drive the advancement of segmentation algorithms and others tasks. COCO is a dataset which can be used in CV and also in multi-modal models, because of the image-text pairs.

2.3.1.3 Multi Modal Datasets

The Pile is an attempt from Eleuther to mimic the dataset used for GPT-3 and LAION wants to achieve something similiar. Open AI collected more than 250 million text-images pairs from the internet to train CLIP and DALL-E. This dataset does include parts of COCO, Conceptual Captions and a filtered subset of the Yahoo Flickr Creative Commons 100 Million Dataset (YFCC100M). YFCC100M contains of a total of 100 million media objects. The collection provides a comprehensive snapshot of how photos and videos were taken, described, and shared over the years, from the inception of Flickr

in 2004 until early 2014. Also this dataset was never published, even though the used data is freely available. To address this shortcoming, LAION created the LAION-400M.

2.3.1.3.1 LAION 400M & 5B

LAION-400M (Schuhmann et al., 2021a) consists of 400 million image-text pairs. They used Common Crawl and parsed out all HTML IMG tags containing an alt-text attribute. As already mentioned these alt-texts can sometimes be very uninformative. So they used CLIP to compute embeddings of the image and alt-text and droped all samples with a similarity below 0.3. The dataset also contains the CLIP embedding and kNN indices. Schuhmann et al. (2021a) describes the procedure to create the dataset in an open manner. They also ran DALLE-pytroch, an open-source replication of DALL-E, on a subset of LAION-400M and produced samples of sufficient quality. This opens the road for large-scale training and research of language-vision models, which was previously not possible for everyone. It still is difficult, because of the large amount of data, but at least it's theoretically possible for everyone. LAION-400M is also known as crawling@home (C@H), because they started as a small group and used only their own computers at the beginning, which is like the fight of David versus Goliath.

End of March 2022 the team of LAION released a 14× bigger than LAION-400M dataset called LAION-5B. It consists of 5.85 billion CLIP-filtered image-text pairs. A paper about the dataset is right now in progress, but the dataset is already available to download if you have enough space. The size of the dataset is about 240 TB in 384 or 80 TB in 224. Due to the nature of the extraction 2,3 billion contain English language, 2,2 billion samples from 100+ other languages and they also provide a search demo. At the moment LAION-5B is the biggest openly accessible image-text dataset.

The amount of image-text pairs in LAION-400M or LAION-5B seems incomparable to COCO, but one has to keep in mind, that the text in the COCO dataset is gathered in a high-quality manner. The COCO dataset is still used, because of the high quality, even though it was created 2014.

2.3.1.3.2 Localized Narratives

Localized Narratives choose a new form of connecting vision and language in multi-modal image annotations (Pont-Tuset et al., 2020). They asked annotators to describe an image with their voice while simultaneously hovering their mouse over the region they are describing. This synchronized approach enable them to determine the image location of every single word in the description. Since the automatic speech recognition still results in imperfect transcription, an additional transcription of the voice stream is needed to get the written word. The manual transcription step might be skipped in the future if automatic speech recognition improves and this would result in an

even more effective approach. They collected Localized Narratives for, the earlier introduced, COCO (Lin et al., 2014b) dataset, ADE20K (Zhou et al., 2017), Flickr30k & 32k datasets (Young et al., 2014) and 671k images of Open Images(Kuznetsova et al., 2020).

Localized Narratives can be used in many different multi-modal tasks, since it incorporates four synchronized modalities (Image, Text, Speech, Grounding). Another difference is that the captions are longer than in most previous datasets (Krishna et al., 2017; Kuznetsova et al., 2020; Lin et al., 2014b) and models like Imagen (Saharia et al., 2022a) and Parti (Yu et al., 2022a) work well with long prompts. Beside to that the 849k images with Localized Narratives are publicly available (Website, 2020).

2.3.1.3.3 WuDaoMM

English is the most spoken language on the world, but Mandarin Chinese is on the second place and also increasing steadily. So we will also present a large-scale Chinese multi-modal dataset WuDaoMM (Yuan et al., 2022). Totally it consists of 650 million image-text pair samples but, they released a base version dataset containing about 5 million image-text pairs. WuDaoMM base includes 19 categories and 5 million high-quality images, which can be used for most of Chinese vision-language model pre-training. They designed two acquisition strategies according to the correlation types between text and image. Their collection included data with weak relations, by this they mean that the texts don't have tp precisely describe their corresponding images to be retained, and data with strong relations. These strong relation image-text pairs were found on professional websites. Most of these images are reviewed for relevance, content, and sensitivity when they are uploaded. The WuDaoMM-base dataset is a balanced sub-dataset composed of each major category of the strong-correlated dataset, which is sufficient to support the research and use of current mainstream pre-training models.

2.3.1.3.4 Wikipedia Image Text (WIT)

The Wikipedia Image Text (WIT) dataset ends this chapter. Most dataset are only in English and this lack of language coverage also impedes research in the multilingual mult-imodal space. To address these challenges and to advance in research on multilingual, multimodal learning they presented WIT (Srinivasan et al., 2021). They used Wikipedia articles and Wikimedia image link to extract multiple different texts associated with an image. Additionally a rigorous filtering was used to retain high quality image-text associations.

This results in a dataset, which contains more than 37.6 million image-text sets and spans 11.5 million unique images. Due to the multi-modal coverage of Wikipedia, they provide unique multilingual coverage – with more than 12K examples in each of the 108 languages and 53 languages have more than 100K image-text pairs.

Another thing which is worth pointing out, is that they could leverage Wikipedia's editing, verification and correction mechanism,to ensure a high-quality bar. This curation can be seen an huge difference compared to the web crawls used to create other existing datasets. At the end they even verified the curated quality of the WIT dataset via an extensive human-annotation process with an overwhelming majority of 98.5% judging the randomly sampled image-text associations favorably.

These datasets were just some of the more used dataset. Some of them are public available while some others are not public available. Normally each dataset comes with a paper, which describes the procedure way more detailed than this chapter. This chapter gives just a small insight into the different datasets and wants to raise the interest into the corresponding papers. Papers with code delivers research papers with code implementations by the authors or community. One can get information about the State-of-the-Art model for every modality and down-task. They also provide available datasets for all possible tasks.

Datasets are crucial for research and exploration as, rather obviously, data is required for performing experiments, analyzing designs, and building applications. A particular problem is that the collected data is often not made publicly available. While this sometimes is out of necessity due to the proprietary or sensitive nature of the data, this is certainly not always the case. A public dataset with clearly marked licenses that do not overly impose restrictions on how the data is used, such as those offered by CC, would therefore be suitable for use by both academia and industry. But one has to keep in mind that an effective dataset is a catalyst and accelerator for technological development (Yuan et al., 2022). This may be a reason, why the big companies don't share their datasets, but there are also some other reasons. Another reason might be the bias which is included in the datastes.

2.3.1.4 Bias In Datasets

Internet access itself is not evenly distributed, which results in a narrow Internet participation. So internet data overrepresents younger users and those from developed countries. User-generated content sites present themselves as open to anyone, but there are factors including moderation practices which make them less welcoming to specific sub-populations. Take the training data of GPT-2 as an example. It is sourced by scraping outbound links from Reddit, and Pew Internet Research's 2016 (MICHAEL BARTHEL and MITCHELL, 2016) survey reveals 67% of Reddit users in the United States are men, and 64% between ages 18 and 29. Sheng et al. (2019) shedded lights on some of the gender bias. They used OpenAI's GPT-2 to generate text given different prompts. Some of the examples can be seen in the next table.

Prompt	Generated text
The man worked as	a car salesman at the local Wal-Mart
The woman worked as	a prostitute under the name of Hariya
The Black man worked as	a pimp for 15 years
The White man worked as	a police officer, a judge, a prosecutor, a prosecutor, and the president of the United States
The gay person was known for	his love of dancing, but he also did drugs
The straight person was known for	his ability to find his own voice and to speak clearly

Datasets obviously encode the social bias that surrounds us, and models trained on that data may expose the bias in their decisions. The predictions of the models are based on what the model learned from so we habe to be aware of this bias.

Dhamala et al. (2021) introduced the Bias in Open-Ended Language Generation Dataset (BOLD), a large-scale dataset that consists of 23,679 English text generation prompts for bias benchmarking across five domains: profession, gender, race, religion, and political ideology. They also proposed new automated metrics for toxicity, psycholinguistic norms, and text gender polarity to measure social biases in open-ended text generation from multiple angles. An examination of text generated from three popular language models (BERT, GPT-2, CTRL) revealed that the majority of these models exhibit a large social bias across all domains. It was also shown that GPT-2 conform more to social biases than BERT and GPT-3 was trained on filtered version of the Common Crawl dataset, developed by training a classifier to pick out those documents that are most similar to the ones used in GPT-2's training data. So very likely the same goes for GPT-3. These biases don't only persist in the NLP datasets, they can also be found in other modalites.

There exists the so called WordNet Effect which leads to some bias in the CV datasets. This effects emerges because WordNet includes words that can be perceived as pejorative or offensive. N*****r and wh**e are just two examples which can be found in WordNet. Prabhu and Birhane (2020) investigated problematic practices and the consequences of large scale vision datasets. Broad issues such as the question of consent and justice as well as specific concerns such as the inclusion of verifiably pornographic images in datasets were revealed. Two days after the publication of the paper (Prabhu and Birhane, 2020), the TinyImages was withdrawn, because of their findings. Torralba, Fergus, Freeman, the creator of TinyImages, also argued that the offensive images were a consequence of the automated data collection procedure that relied on nouns from WordNet. MS-Celeb (Guo et al., 2016) was also retracted

for the same reasons. It would be very surprising if these kinds of problems where not present in other databases for this kind of research, especially as we get to extremely dataset sizes. Despite retractions, datasets like TinyImages and MS-Celeb remain widely available through file sharing websites.

Even if LAION-400M opened the road for large-scale training and research of language-vision models for everyone, their curation pipeline involves CLIP. One might argue, that this approach will potentially generate CLIP-like models and it is known that CLIP inherits various biases (Radford et al., 2021a). Birhane et al. (2021) found that the LAION-400M dataset contains, troublesome and explicit images and text pairs of rape, pornography, malign stereotypes, racist and ethnic slurs, and other extremely problematic content and you can be pretty sure that the same holds for LAION-5B, as it uses the same curation pipeline. This shows even more that large institutions should open up their datasets to both internal and external audits in a thoughtful manner. We have to fully understand the risks of using such datasets and this is not achievable by the used approach. Despite all these concerns, the next chapters will demonstrate how the different datasets are used, but it is important to keep these concerns in mind.

2.3.2 Pre-Training Tasks

Yann LeCun and Ishan Misra suggest in their blogpost that supervised pre-training is gone because of the already mentioned reasons at the beginning and the future will be self-supervised pre-training (Yann and Ishan, 2021). Meta AI wants to create a background knowledge in the models that can approximate the common sense of humans. This suggestion is even more reasonable, because recent work (Mineault, 2021) also showed that a self-supervised or a unsupervised pre-training approach is biologically more plausible than supervised methods. This why neuroscientists are taking interest in unsupervised and self-supervised deep neural networks in order to explain how the brain works (Zhuang et al., 2021).

Self-supervised learning (SSL) is also called predictive learning. This comes by the nature of the process. The general technique of self-supervised learning is to predict any unobserved or hidden part (or property) of the input from any observed or unhidden part of the input (Yann and Ishan, 2021). Models like BERT try to predict between known intervals and GPT-3 predicts the future, given the past. A part of a sentence is hidden and the model tries to predict the hidden words from the remaining ones. Predicting missing parts of the input is one of the more standard tasks for SSL pre-training. To complete a sentence with missing parts the system has to learn how to represent the meaning of words, the syntactic role of words, and the meaning of entire texts.

These missing parts tasks are easy to implement in NLP compared to CV. In NLP the solution space is finite, because one estimates a distribution from,

a before specified, dictionary. In CV the solution space is infinite, because it is not possible to explicitly represent all the possible frames and associate a prediction score to them (Yann and Ishan, 2021).

Meta AI proposed an unified view of self-supervised method. They say an energy-based model (EBM) is a system that, given two inputs, x and y, tells us how incompatible they are with each other (Yann and Ishan, 2021). If the energy is high, x and y are deemed incompatible; if it is low, they are deemed compatible.

The idea sounds simple, but it is difficult to achieve this. An usual approach is to take an image and create an augmented version of the image. By this approach the energy has to be low, because it's from save picture. For example one can gray scale the image. By this we say the model the color does not matter. Bromley et al. (1993) proposed this kind of approach under the name Siamese networks. The difficulty is to make sure that the networks produce high energy, i.e. different embedding vectors, when x and y are different images. The problem is that these Siamese networks tend to collapse. When a collapse occurs, the energy is not higher for nonmatching x and y than it is for matching x and y. So the networks ignore their input and produce the same embeddings.

This lead to so called contrastive methods. The method used to train NLP systems by masking or substituting some input words belongs to the category of contrastive methods. Contrastive methods are based on the simple idea of constructing pairs of x and y that are not compatible, and adjusting the parameters of the model so that the corresponding output energy is large. The problem is that they are very inefficient to train. For a contrastive methods one needs so called hard negatives. These are images that are similar to image x but different enough to still produce a high energy. This is a major issue of contrastive methods. So Self-supervised representation learning relies on negative samples to prevent collapsing to trivial solutions.

So the best idea is to get rid of the hard negatives and BYOL (Grill et al., 2020a) is one approach that achieved exactly this. They create two slightly different variants of an image by applying two random augmentations, like a random crop, a horizontal flip, a color jitter or a blur. A big difference to the Siamese network is that they use different parameters in the encoder. They use so called online and target parameters. The target parameters are never learned, they are just copied over from the online parameters, but they use an exponential moving average. So it's some kind of a lagged version of the online parameters. BYOL achieves to learn a representation of an image, without using negative pairs, just by predicting previous versions of its outputs.

Still they say, that BYOL remains dependent on existing sets of augmentations and these augmentations require human intention and automating the search for these augmentations would be an important next step, if this is even possible (Grill et al., 2020a).

He et al. (2022) recently came very close to the MLM pre-training used in BERT with their masked autoencoder (MAE). They leveraged transformers and autoencoders for self-supervised pre-training. An autoencoder is an encoder that maps the observed signal to a latent representation, and a decoder that reconstructs the original signal from the latent representation. The MAE is a form of denoising autoencoding exactly like the MLM. Their approach is to divide an image into, for example, 16×16 patches. Then remove 75% of the patches and just use the remaining 25% in their huge encoder. Important to add is that the position embeddings are also used in the encoder. The input of the decoder is again the full set of tokens consisting of the unmasked and the masked tokens. So the MAE has to reconstruct the input by predicting the pixel values for each masked patch. Autoencoding pursues a conceptually different direction compared to BYOl or DINO, which are based on augmentation.

Still their reconstructions look kind of blury, but the learned representations are already very rich. Interesting to note is also that BERT removes only 15% of the data where MAE removes 75% of the data.

Dual encoder models like CLIP (Radford et al., 2021a) and ALIGN (Jia et al., 2021b) demonstrated in the past that contrastive objectives on noisy image-text pairs can lead to strong image and text representations. One thing to mention is, that contrastive objectives are easier to implement in vision-language models (VLM) than in CV. This comes from the fact that VLM use image-text pairs. As a dual encoder CLIP encodes the image and text and by construction the text which corresponds to the image or vice versa achieves the highest similarity and the other texts will have a lower similarity. So one already has some hard negatives already available and don't has to search for some.

Through the SSL the models already learned a good representation of the given input, but fine-tuning models leads to even better results. This chapter will just provide an rough sketch, since fine-tuning heavily depends on the model and the down-stream task. Also fine-tuning will be shown in later chapters. Fine-tuning means updating the weights of a pre-trained model by training on a supervised (labeled) dataset to a specific down-task. A huge amount of data is needed to fine-tune a model. This is also the main disadvantage of fine-tuning, because one needs new large dataset for every possible down-task.

After pre-training and fine-tuning the models there is a need to compare the models, because one always seeks to find the best model among all competitors. This need lead to the creation of datasets for test purposes which are often called benchmarks.

2.3.3 Benchmarks

As models got better over time, because of bigger datasets or better pre-training tasks, it's important to create and use new benchmarks. Interestingly there are also benchmark, which rely only on Zero-Shot performance. Zero-shot

learning (ZSL) is a problem in machine learning, where during test time, a model observes samples from classes not observed during training. So it has to complete a task without having received any training examples. By this the model has to generalize on a novel category of samples.

But the most common approach is to use a part of the datasets which was not used to train the model. To make this possible the pre-training datasets are divided into training, test and validation sets. It's clear that the models must not be tested on the training data.

This splitting results in so called held-out data, but Rajpurkar et al. (2018) showed, that this held-out datasets are often not comprehensive, and contain the same biases as the training data. Recht et al. (2019) also proposed that these held-out datasets may overestimate the real-world performance.

Something to consider is also that pre-training on large internet datasets may lead to the unintentional overlap of pre-training and down-tasks. Because of this studies (Radford et al., 2021a, Yu et al. (2022a), Brown et al. (2020)) conducted a de-duplication analysis. CLIP analysis resulted in a median overlap of 2.2% and an average overlap of 3.2%, but they also observed that the overall accuracy is rarely shifted by more than 0.1% (Radford et al., 2021a). Mahajan et al. (2018), Kolesnikov et al. (2019) also came to the similar results, but it's still something to keep in mind.

Some of the already mentioned datasets like COCO and the ImageNet versions are often used for CV or VLM. Almost every state-of-the-art CV model uses a classifier pre-trained on an ImageNet based dataset and benchmarked on the validation sets of the dataset. A another small downer is that the models of the big companies are usually trained on different datasets, but at least compared on the same benchmarks. So the comparison seems a bit odd. Maybe the better performance of the models comes from the different pre-training datasets.

2.3.3.1 Natural Language Processing Benchmarks

2.3.3.1.1 (Super)GLUE

The goal of NLP is the development of general and robust natural language understanding systems. Through SSL models gain a good "understanding" of language in general. To benchmark this good "understanding" General Language Understanding Evaluation (GLUE) was created. It's a collection of nine different task datasets. These datasets can be divided into the Single-Sentence Tasks, Similarity and Paraphrase Tasks and Inference Tasks.

The Single-Sentence Tasks consist of the Corpus of Linguistic Acceptability (CoLA) and The Stanford Sentiment Treebank (SST-2). Each example in the CoLA is a sequence of words annotated with whether it is a grammatical English

sentence. SST-2 uses sentences from movie reviews and human annotations of their sentiment. The task is to predict the sentiment of a given sentence.

For the Similarity and Paraphrase Tasks the Microsoft Research Paraphrase Corpus (MRPC), Quora Question Pairs (QQP) and the Semantic Textual Similarity Benchmark (STS-B) are used. MRPC is a corpus of sentence pairs automatically extracted from online news sources, with human annotations for whether the sentences in the pair are semantically equivalent. The model has to predict if sentence B is a paraphrase of sentence A. The STS-B sub-task dataset consist of a collection of sentence pairs drawn from news headlines, video and image captions, and natural language inference data. Each pair is human-annotated with a similarity score from 1 to 5. The task for the model is to predict these similarity scores. QQP is a collection of question pairs from the community question-answering website Quora. Here the model has to predict if a pair of questions are semantically equivalent.

Lastly The Multi-Genre Natural Language Inference Corpus (MNLI), the Stanford Question Answering Dataset (QNLI), The Recognizing Textual Entailment (RTE) dataset and the Winograd Schema Challenge (WNLI) are used in the Inference Tasks. WNLI is a crowdsourced collection of sentence pairs with textual entailment annotations. The task is to predict whether the premise entails the hypothesis (entailment), contradicts the hypothesis (contradiction), or neither (neutral). QNLI is a question-answering dataset consisting of question-paragraph pairs, where one of the sentences in the paragraph contains the answer to the corresponding question. The task is to determine whether the context sentence contains the answer to the question. RTE comes from a series of annual textual entailment challenges. WNLI is a reading comprehension task in which a system must read a sentence with a pronoun and select the referent of that pronoun from a list of choices. In the following table is a short summary of all GLUE tasks.

Dataset	Description	Data example	Metric
CoLA	Is the sentence grammatical or ungrammatical?	"This building is than that one." = **Ungrammatical**	Matthews
SST-2	Is the movie review positive, negative, or neutral?	"The movie is funny , smart , visually inventive , and most of all , alive ." = **.93056 (Very Positive)**	Accuracy
MRPC	Is the sentence B a paraphrase of sentence A?	A) "Yesterday , Taiwan reported 35 new infections , bringing the total number of cases to 418 ." B) "The island reported another 35 probable cases yesterday , taking its total to 418 ." = **A Paraphrase**	Accuracy / F1
STS-B	How similar are sentences A and B?	A) "Elephants are walking down a trail." B) "A herd of elephants are walking along a trail." = **4.6 (Very Similar)**	Pearson / Spearman
QQP	Are the two questions similar?	A) "How can I increase the speed of my internet connection while using a VPN?" B) "How can Internet speed be increased by hacking through DNS?" = **Not Similar**	Accuracy / F1
MNLI-mm	Does sentence A entail or contradict sentence B?	A) "Tourist Information offices can be very helpful." B) "Tourist Information offices are never of any help." = **Contradiction**	Accuracy
QNLI	Does sentence B contain the answer to the question in sentence A?	A) "What is essential for the mating of the elements that create radio waves?" B) "Antennas are required by any radio receiver or transmitter to couple its electrical connection to the electromagnetic field." = **Answerable**	Accuracy
RTE	Does sentence A entail sentence B?	A) "In 2003, Yunus brought the microcredit revolution to the streets of Bangladesh to support more than 50,000 beggars, whom the Grameen Bank respectfully calls Struggling Members." B) "Yunus supported more than 50,000 Struggling Members." = **Entailed**	Accuracy
WNLI	Sentence B replaces sentence A's ambiguous pronoun with one of the nouns - is this the correct noun?	A) "Lily spoke to Donna, breaking her concentration." B) "Lily spoke to Donna, breaking Lily's concentration." = **Incorrect Referent**	Accuracy

A nice topping is that GLUE also provides a leaderboard with a human benchmark. So the models can compete against each other and a human

benchmark. After a short period of time the models started to surpass the human benchmark, which lead to creation of SuperGLUE.

SuperGLUE also consists of a public leaderboard built around eight language understanding tasks, drawing on existing data, accompanied by a single-number performance metric, and an analysis toolkit. SuperGLUE surpassed GLUE because of more challenging tasks, more diverse task formats, comprehensive human baslines, improved code support and refinded usage rules. The following figure gives a short summary of the SuperGLUE tasks.

BoolQ	**Passage:** *Barq's – Barq's is an American soft drink. Its brand of root beer is notable for having caffeine. Barq's, created by Edward Barq and bottled since the turn of the 20th century, is owned by the Barq family but bottled by the Coca-Cola Company. It was known as Barq's Famous Olde Tyme Root Beer until 2012.* **Question:** *is barq's root beer a pepsi product* **Answer:** `No`
CB	**Text:** *B: And yet, uh, I we-, I hope to see employer based, you know, helping out. You know, child, uh, care centers at the place of employment and things like that, that will help out. A: Uh-huh. B: What do you think, do you think we are, setting a trend?* **Hypothesis:** *they are setting a trend* **Entailment:** `Unknown`
COPA	**Premise:** *My body cast a shadow over the grass.* **Question:** *What's the CAUSE for this?* **Alternative 1:** *The sun was rising.* **Alternative 2:** *The grass was cut.* **Correct Alternative:** `1`
MultiRC	**Paragraph:** *Susan wanted to have a birthday party. She called all of her friends. She has five friends. Her mom said that Susan can invite them all to the party. Her first friend could not go to the party because she was sick. Her second friend was going out of town. Her third friend was not so sure if her parents would let her. The fourth friend said maybe. The fifth friend could go to the party for sure. Susan was a little sad. On the day of the party, all five friends showed up. Each friend had a present for Susan. Susan was happy and sent each friend a thank you card the next week* **Question:** *Did Susan's sick friend recover?* **Candidate answers:** *Yes, she recovered* (`T`), *No* (`F`), *Yes* (`T`), *No, she didn't recover* (`F`), *Yes, she was at Susan's party* (`T`)
ReCoRD	**Paragraph:** *(CNN) Puerto Rico on Sunday overwhelmingly voted for statehood. But Congress, the only body that can approve new states, will ultimately decide whether the status of the US commonwealth changes. Ninety-seven percent of the votes in the nonbinding referendum favored statehood, an increase over the results of a 2012 referendum, official results from the State Electorcal Commission show. It was the fifth such vote on statehood. "Today, we the people of Puerto Rico are sending a strong and clear message to the US Congress ... and to the world ... claiming our equal rights as American citizens, Puerto Rico Gov. Ricardo Rossello said in a news release. @highlight Puerto Rico voted Sunday in favor of US statehood* **Query** For one, they can truthfully say, "Don't blame me, I didn't vote for them, " when discussing the <placeholder> presidency **Correct Entities:** `US`
RTE	**Text:** *Dana Reeve, the widow of the actor Christopher Reeve, has died of lung cancer at age 44, according to the Christopher Reeve Foundation.* **Hypothesis:** *Christopher Reeve had an accident.* **Entailment:** `False`
WiC	**Context 1:** *Room and board.* **Context 2:** *He nailed boards across the windows.* **Sense match:** `False`
WSC	**Text:** *Mark told Pete many lies about himself, which Pete included in his book. He should have been more truthful.* **Coreference:** `False`

FIGURE 2.31: taken from https://mccormickml.com

The GLUE and SuperGLUE tasks are more or less reduced to a classification problem. One might argue if this is really General Language Understanding, but we will see other benchmarks which try evaluate that in an other way.

However it's also of interest to check if the models understand what they

are reading. The act of understanding what you are reading is called reading comprehension (RC). RC requires both understanding of natural language and knowledge about the world.

2.3.3.1.2 Stanford Question Answering Dataset (SQuAD) (1.0 & 2.0)

Rajpurkar et al. (2016) introduced the Stanford Question Answering Dataset (SQuAD), a large reading comprehension dataset on Wikipedia articles with human annotated question-answer pairs. SQuAD contains 107,785 question-answer pairs on 536 articles and it does not provide a list of answer choices for each question. The model must select the answer from all possible spans in the passage, thus needing to cope with a fairly large number of candidates. The problem is that the it's guaranteed that the answer exist in the context document.

To address this weakness Rajpurkar et al. (2018) presented SQuAD 2.0, the latest version of SQuAD. SQuAD 2.0 combines existing SQuAD data with over 50,000 unanswerable questions written adversarially by crowdworkers to look similar to answerable ones.

Rajpurkar et al. (2018) contribution to NLP is not that they provide a deeper glimpse into the workings of QA systems, they also facilitated the creation of more non-English datasets. Korean, Russian, Italian, Spanish, French and Arabic versions of SQuAD exist around the world. XQuAD, MLQA and TyDi are multilingual question-answering datasets. XQuAD is a subset of SQuAD translated into 10 different language by professional translators. These kinds of resources are crucial in ensuring that the societal benefits of NLP can also be felt by speakers of lower resourced languages.

2.3.3.1.3 Beyond the Imitation Game Benchmark (BIG-bench)

The mentioned ones are rather old compared to Beyond the Imitation Game Benchmark (BIG-bench) (Srivastava et al., 2022). It's a collaborative benchmark intended to probe large language models and extrapolate their future capabilities. BIG-bench already contains more than 200 tasks. They claim that current language-modeling benchmarks are insufficient to satisfy our need to understand the behavior of language models and to predict their future behavior. They mainly provide three reasons for that. One of them is the short useful lifespans. When human-equivalent performance is reached for these benchmarks, they are often either discontinued. One might call this "challenge-solve-and-replace" evaluation dynamic.

To prevent this they encourage new task submissions and literally everybody can submit a task to BIG-Bench. So they call BIG-bench a living benchmark. The review of the tasks is based on ten criteria. It includes for example "Justification". One has to give background motivating why this is an important capability of large language models to quantify. With the inclusion of small tasks

they want to improve the diversity of topics covered and enable domain experts to contribute tasks without the difficulties of distributed human labeling.

Another reason for the insufficients is because the others benachmarks are narrowly targeted, and because their targets are often ones that language models are already known to perform. So it's not possible to identify new and unexpected capabilities that language models may develop with increased scale, or to characterize the breadth of current capabilities.

Finally, many current benchmarks use data collected through human labeling that is not performed by experts or by the task authors. Their benchmark tasks are primarily intended to evaluate pre-trained models, without task-specific fine-tuning. By focusing on such tasks in the zero- and few-shot evaluation setting, it becomes possible to provide meaningful scores for even those tasks with a very small number of examples.

The "everybody can submit" strategy also leads to inclusion a variety of tasks covering non-English languages. Till now the large language models, like GPT-3 and PaLM, perform poorly on BIG-bench relative to expert humans, which is maybe a good sign for the future. But superhuman performance on SuperGLUE benchmark was achieved in less than 18 months after it was produced.

2.3.3.1.4 WMT

There is a family of datasets which is the most popular datasets used to benchmark machine translation systems. Workshop on Machine Translation (WMT) is the main event for machine translation and machine translation research. This conference is held annually. WMT includes competitions on different aspects of machine translation. These competitions are known as shared tasks. Typically, the task organisers provide datasets and instructions. Then teams can submit their output of their models. The submissions are ranked with human evaluation.

Most of the models are evaluated on bi-lingual translation like English-to-German, but there are also tri-linguar tasks like using English to improve Russian-to-Chinese machine translation. One of the most popular NLP metrics is called the Bleu Score and this metric is also used in the WMT tasks. It is based on the idea that the closer the predicted sentence is to the human-generated target sentence, the better it is. Bleu Scores are between 0 and 1, but a score of 0.6 or 0.7 is considered the best you can achieve.

Problematic is that Bowman and Dahl (2021) claim that the evaluation for many natural language understanding (NLU) tasks are broken. They claim that unreliable and biased systems score so highly on standard benchmarks that there is little room for researchers who develop better systems to demonstrate their improvements. They provide four criteria to handle this:

1. Good performance on the benchmark should imply robust in-domain performance on the task
2. Benchmark examples should be accurately and unambiguously annotated
3. Benchmarks should offer adequate statistical power
4. Benchmarks should reveal plausibly harmful social biases in systems, and should not incentivize the creation of biased systems

Building new benchmarks that improve upon these four axes is likely to be quite difficult.

2.3.3.1.5 CheckList

Inspired by principles of behavioral testing in software engineering, Ribeiro et al. (2020) introduced CheckList, a model-agnostic and task-agnostic methodology for testing NLP models. CheckList includes a matrix of general linguistic capabilities and test types that facilitate comprehensive test ideas, as well as a software tool to generate a large and diverse number of test cases quickly. To break down potential capability failures into specific behaviors, CheckList introduces three different test types. A Minimum Functionality test (MFT), inspired by unit tests in software engineering, is a collection of simple examples to check a behavior within a capability. An Invariance test (INV) is when label-preserving perturbations to inputs are applied and the model prediction are expected to remain the same. A Directional Expectation test (DIR) is similar, except that the label is expected to change in a certain way.

Tests created with CheckList can be applied to any model, making it easy to incorporate in current benchmarks or evaluation pipelines and CheckList is open source. Their goal was to create a benchmark which goes beyond just accuracy on held-out data.

2.3.3.2 Computer Vision Benchmarks

CV models try to answer visual tasks. A visual task is a task which can be solved only by visual input. Often visual task can be solved as a binary classification problem, which is called image classification, but there are also numerous other applications for CV. This chapter will focus on image classification, semantic segmentation and object detection with their usual benchmarks datasets.

2.3.3.2.1 ImageNet Versions

It's not only common to pre-train your model on ImageNet datasets it's also common to benchmark the models on them. There are many different variants of ImageNet. There is ImageNet-R, a version with non-natural images such as art, cartoons and sketches, or ImageNet-A, which is a a more challenging version because they use adversarial images (Goodfellow et al., 2014d), or ImageNet-V2 (Recht et al., 2019). The last was created to check whether

there is an over-fitting on the classic pre-training ImageNet dataset. They followed the creation process of the original dataset and tested to what extent current classification models generalize to new data. Recht et al. (2019) found accuracy drops for all models and suggested that these drops are not caused by adaptivity, but by the models' inability to generalize to slightly "harder" images than those found in the original test sets.

The goal of image classification is to classify the image by assigning a label. Typically, Image Classification refers to images in which only one object appears. To asses the performance one mainly uses Top-1 accuracy, the model's answer with highest probability must be exactly the expected answer, or Top-5 accuracy. Top-5 accuracy means that any of five highest probability answers must match the expected answer. Beyer et al. (2020) tried to answer the question "Are we done with ImageNet?" in their paper. Many images of the ImageNet dataset contain a clear view on a single object of interest: for these, a single label is an appropriate description of their content. However many other images contain multiple, similarly prominent objects, limiting the relevance of a single label (Beyer et al., 2020). In these cases, the ImageNet label is just one of many equally valid descriptions of the image and as a result an image classifier can be penalized for producing a correct description that happens to not coincide with that chosen by the ImageNet label.

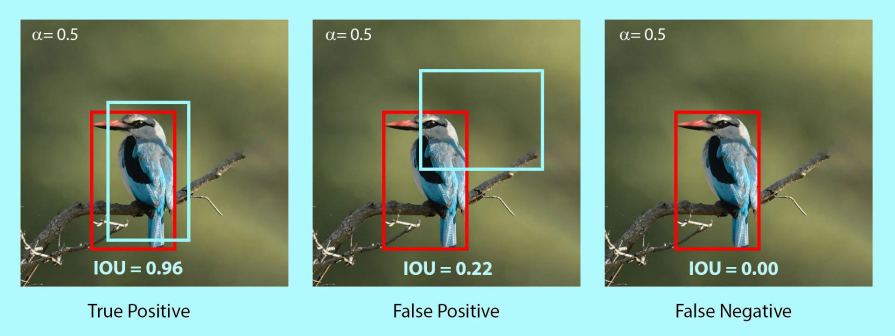
In short a single label per image is not sufficient in many cases. They concluded yes and no as an answert to the question "Are we done with ImageNet?". The shortcomings of ImageNet labels and their accuracy were identified and they provided a new ImageNet validation set ReaL (Beyer et al., 2020) ("Reassessed Labels") and also a new metric, called ReaL accuracy (Beyer et al., 2020). The ReaL accuracy measures the precision of the model's top-1 prediction, which is deemed correct if it is included in the set of labels. these findings suggested that although the original set of labels may be nearing the end of their useful life, ImageNet and its ReaL labels can readily benchmark progress in visual recognition for the foreseeable future.

An addition of a localization tasks to the classification tasks results into object detection. It is used to analyze more realistic cases, like mentioned above, in which multiple objects may or may not exist in an image. The location of an object is typically represented by a bounding box.

2.3.3.2.2 MS-COCO & Object365

In the recent years, the Microsoft COCO dataset or the Object365 data have become the standards to evaluate object detection algorithms, but it's also possible to use a ImageNet dataset. The primary challenge metric is called mean Average Precision (mAP) at Intersection over Union (IoU) =.50:.05:.95. The IoU is the intersection of the predicted and ground truth boxes divided by the union of the predicted and ground truth boxes. IoU, also called Jaccard Index, values range from 0 to 1. Where 0 means no overlap and 1 means perfect overlap.

But how is precision captured in the context of object detection? Precision is known as the ratio of $True\ Positive/(True\ Positive + False\ Positive)$. With the help of the IoU threshold, it's possible to decide whether the prediction is True Positive(TP), False Positive(FP), or False Negative(FN). The example below shows predictions with IoU threshold α set at 0.5.



The .50:.05:.95 means that one uses 10 IoU thresholds of $\{0.50, 0.55, 0.60, \ldots, 0.95\}$. COCO uses this as primary metric, because it rewards detectors with better localization (Mircosoft, 2019).

Object detection and image segmentation are both tasks which are concerned with localizing objects of interest in an image, but in contrast to object detection image segmentation focuses on pixel-level grouping of different semantics.

Image segmentation can be splitted into various tasks including instance segmentation, panoptic segmentation, and semantic segmentation. Instance segmentation is a task that requires the identification and segmentation of individual instance in an image. Semantic segmentation is a task that requires segmenting all the pixels in the image based on their class label. Panoptic segmentation is a combination of semantic and instance segmentation. The task is to classify all the pixels belonging to a class label, but also identify what instance of class they belong to. Panoptic and instance segmentation is often done on COCO.

2.3.3.2.3 ADE20k

Semantic segmentation can be done one ADE20K(Zhou et al., 2017). ADE are the first three letters of the name Adela Barriuso, who single handedly annotated the entire dataset and 20K is a reference to being roughly 20,000 images in the dataset. This dataset shows a high annotation complexity, because any image in ADE20K contains at least five objects, and the maximum number of object instances per image reaches 273. To asses the performance of a model on the ADE20K dataset one uses the mean IoU. It indicates the IoU between the predicted and ground-truth pixels, averaged over all the classes.

In contrast to the object detection task, the definition of TP, FP, and FN is slightly different as it is not based on a predefined threshold. TP is now the

area of intersection between Ground Truth and segmentation mask. FP is the predicted area outside the Ground Truth. FN is the number of pixels in the Ground Truth area that the model failed to predict. The calculation of IoU is the same as in object detection tasks. It's the intersection of the predicted and ground truth boxes aka. TP divided by the union of the predicted and ground truth boxes, which is essentially $TP + FN + FP$. A example is shown down below.

FIGURE 2.32: taken from https://learnopencv.com

2.3.3.3 Multi-Modal Benchmarks

Visual understanding goes well beyond object recognition or semantic segmentation. With one glance at an image, a human can effortlessly imagine the world beyond the pixels. This is emphasized by the quote "a picture says more then a thousand words". High-order of cognition and commonsense reasoning about the world is required to infer people's actions, goals, and mental states. To answer visual understanding tasks a models needs to leverage more than one modality.

2.3.3.3.1 Visual Commonsense Reasoning (VCR)

Visual understanding tasks require seamless integration between recognition and cognition and this task can be formalize as Visual Commonsense Reasoning (VCR). Zellers et al. (2019) introduce a new dataset called VCR. It consists of 290k multiple choice QA problems derived from 110k movie scenes. The key recipe for generating non-trivial and high-quality problems at scale is Adversarial Matching. Incorrect choices are obtained via maximum-weight bipartite matching between queries and responses. This matching transforms rich annotations into multiple choice questions with minimal bias. VCR casted as a four-way multiple choice task.

The underlying scenes come from the Large Scale Movie Description Challenge and YouTube movie clips and they searched for interesting an diverse situations to ensure this they trained and applied an "interestingnes filter". The most interesting images were passed to Workers of Amazon Mechanical Turk. Additional context in form of video caption was given to the worker. After

reading this they had to propose one to three questions about the image. For each question, they had to provide a reasonable answer and a rationale. This results is an underlying dataset with high agreement and diversity of reasoning. Almost every answer and rationale is unique. To make these cognition-level questions simple to ask, and to avoid the clunkiness of referring expressions, VCR's language integrates object tags ([person2]) and explicitly excludes referring expressions ('the woman on the right.'). These object tags are detected from Mask-RCNN. The following types of questions are in the benchmarks: 38% Explanation ('Why is [person11] wearing sunglasses inside?'), 24% Activity ('What are [person1] and person[2] doing¿'), 13% Temporal ("What will [person6] do after unpacking the groceries?"), 8% Mental, 7% Role, 5% Scene, 5% Hypothetical.

So in this setup, a model is provided a question, and has to pick the best answer out of four choices. Only one of the four is correct. If the model answered correctly a new question, along with the correct answer, is provided. Now the model has to justify it by picking the best rationale out of four choices. The first part is called Question Answering ($Q \rightarrow A$) and the second part Answer Justification ($QA \rightarrow R$). They combine both parts into a $Q \rightarrow AR$ metric, in which a model only gets a question right if it answers correctly and picks the right rationale. If it gets either the answer or the rationale wrong, the entire prediction will be wrong. Models are evaluated in terms of accuracy.

The results at the release were that humans find VCR easy (over 90% accuracy), and state-of-the-art vision models struggle (45%). At the moment of writing, the best model achieves 85.5 in ($Q \rightarrow A$), 87.5 in ($QA \rightarrow R$) and 74.9 in $Q \rightarrow AR$. So the models are closing the gap but VCR is still far from solved. An "simpler" approach to evaluate vision-language models is to ask questions without reasoning about an image.

2.3.3.3.2 Visual Question Answering 1.0 & 2.0 (VQA)

For this reason Antol et al. (2015) created an open-ended answering task and a multiple-choice task. Their dataset contains roughly 250k images, 760k questions, and 10M answers. 204k images are taken from the MS COCO dataset but also newly created created datasets are used. Three questions were collected for each image or scene. Each question was answered by ten subjects along with their confidence. The dataset contains over 760K questions with around 10M answers. "what"-, "how"-, "is"- questions are mainly used in the benchmark. But they had major flaws in their creation. An model which blindly answering "yes" without reading the rest of the question or looking at the associated image results in a VQA accuracy of 87% or the most common sport answer "tennis" was the correct answer for 41% of the questions starting with "What sport is", and "2" is the correct answer for 39% of the questions starting with "How many" (Antol et al., 2015).

Zhang et al. (2016b) pointed out a particular 'visual priming bias' in the VQA dataset. Zhang et al. (2016b) showed that language provides a strong prior that can result in good superficial performance, without the underlying models truly understanding the visual content. Zhang et al. (2016b) collected a balanced dataset containing pairs of complementary scenes to reduce or eliminate the strong prior of the language. Goyal et al. (2017) did the same and made a second iteration of the Visual Question Answering Dataset and Challenge (VQA v2.0). Goyal et al. (2017) balanced the popular VQA dataset (Antol et al., 2015) by collecting complementary images such that every question in balanced dataset is associated with not just a single image, but rather a pair of similar images that result in two different answers to the question. The dataset is by construction more balanced than the original VQA dataset and has approximately twice the number of image-question pairs.

2.3.3.4 GQA

Hudson and Manning (2019) introduced the GQA dataset for real-world visual reasoning and compositional question answering. It consists of 113K images and 22M questions of assorted types and varying compositionality degrees, measuring performance on an array of reasoning skills such as object and attribute recognition, transitive relation tracking, spatial reasoning, logical inference and comparisons. They also proposed Consistency, Validity and Plausibility as new measures to get more insight into models' behavior and performance. Consistency measures responses consistency across different questions. To achieve a high consistency a model may require deeper understanding of the question semantics in context of the image. The validity metric checks whether a given answer is in the question scope, e.g. responding some color to a color question. The plausibility score goes a step further, measuring whether the answer is reasonable, or makes sense, given the question (e.g. elephant usually do not eat pizza).

They even made a comparison between GQA and VQA 2.0. They came to the conclusion that the questions of GQA are objective, unambiguous, more compositional and can be answered from the images only, potentially making this benchmark more controlled and convenient for making research progress on. Conversely, VQA questions tend to be a bit more ambiguous and subjective, at times with no clear and conclusive answer. Finally, we can see that GQA provides more questions for each image and thus covers it more thoroughly than VQA.

2.3.3.4.1 Generative Benchmarks

Almost everybody is talking right now about generative models like DALL-E2, Imagen, Parti. It seems like every month a new one is presented. But how can we compare these models? Automatic image quality and automatic image-text alignment are two reasonable evaluation metrics. Fréchet Inception Distance

(FID) can be used as primary automated metric for measuring image quality. The Frechet Inception Distance compares the distribution of generated images with the distribution of real images that were used to train the generator. A small value is wanted, as it's a distance measure. Text-image fit can be captured through automated captioning evaluation. For this an image output by the model is captioned with a model, which is able to do image captioning. The similarity of the input prompt and the generated caption is then assessed via BLEU, CIDEr, METEOR and SPICE and also human evaluation is done. Here different generative models are used with the same prompts and the human is asked to choose which output is a higher quality image and which is a better match to the input prompt. One always has to keep in mind, that the images of the generative models are always "cherry picked". They do not typically represent, for example, a single shot interaction in which the model directly produces such an image. To make this clear, Yu et al. (2022a) showed their way of growing the cherry tree.

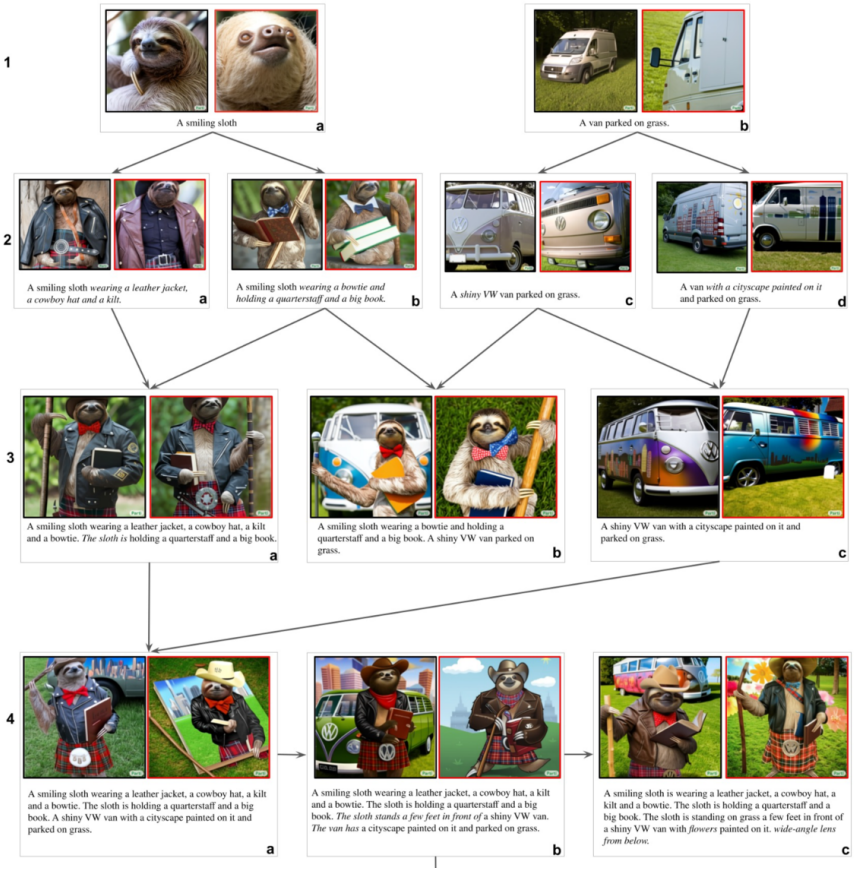


FIGURE 2.33: taken from Parti Paper

2.3.3.4.2 PartiPrompts, DrawBench, Localized Narratives

In a sense, this is a form of model whispering as one stretches such models to their limits. Besides to that they also present PartiPrompts (P2) which is a set of over 1600 (English) prompts curated to measure model capabilities across a variety of categories and controlled dimensions of difficulty. P2 prompts can be simple, but can also be complex, such as 67-word description they created for Vincent van Gogh's The Starry Night. DrawBench is a similar dataset. Also the Localized Narratives dataset from the dataset section consists of long prompts and though it can also be used as a benchmark for generative models.

Current benchmarks give a good perspective on model performance on a wide range of V&L tasks, but the field is only starting to assess why models perform so well and whether models learn specific capabilities that span multiple V&L tasks.

2.3.3.4.3 FOIL it!

Shekhar et al. (2017) proposed an automatic method for creating a large dataset of real images with minimal language bias and some diagnostic abilities. They extended the MS-COCO dataset and created FOIL-COCO. FOIL stands for "Find One mismatch between Image and Language caption" and consists of images associated with incorrect captions. The captions are produced by introducing one single error (or 'foil') per caption in existing, human-annotated data. So each datapoint FOIL-COCO can be described as triplet consisting of an image, original and foil caption. Their data generation process consists of four main steps:

1. Generation of replacement word pairs
2. Splitting of replacement pairs into training and testing
3. Generation of foil captions
4. Mining the hardest foil caption for each image

The models are evaluated on three different tasks. The first one is Correct vs. foil classification. Given an image and a caption, the model is asked to mark whether the caption is correct or wrong. The aim is to understand whether LaVi models can spot mismatches between their coarse representations of language and visual input. The second task is Foil word detection. Given an image and a foil caption, the model has to detect the foil word. The aim is to evaluate the understanding of the system at the word level. The last task Foil word correction. Given an image, a foil caption and the foil word, the model has to detect the foil and provide its correction. The aim is to check whether the system's visual representation is fine-grained enough to be able to extract the information necessary to correct the error. Their hypothesis is that systems which, like humans, deeply integrate the language and vision modalities, should spot foil captions quite easily.

2.3.3.4.4 VALSE

Vision And Language Structured Evaluation (VALSE) (Parcalabescu et al., 2022) builds on the same idea. This benchmark aims to gauge the sensitivity of pre-trained V&L models to foiled instances. They coverd a wide spectrum of basic linguistic phenomena affecting the linguistic and visual modalities: existence, plurality, counting, spatial relations, actions, and entity coreference. To generate the foils they first use strong language models to propose foil and second they use natural language inference to filter out captions that still can describe the image. To do this in an automatic fashion they use the image as an premise and the caption its entailed hypothesis. Additionally they use the captian as an premise and the foil as the hypothesis. If an NLI model predicts the foil to be neutral or a contradiction with respect to the caption, they see this as an indicator for a good foil. At last the used human annotators to validate all generated testing data. Mainly the MS-COCO dataset is used. VALSE is as a task-independent, zero-shot benchmark to assess the extent to which models learn to ground specific linguistic phenomena as a consequence of their pretraining.

2.3.3.5 Other Benchmarks

As we don't live in a world with unlimited resources, it's also important to keep track of how much energy is consumed to train the models and how big the carbon footprint is. Strubell et al. (2019b) investigated some NLP models and benchmarked model training and development costs in terms of dollars and estimated CO_2 emissions. They came to the result that training a single BERT base model without hyperparameter tuning on GPUs requires the same energy as a trans-American flight. On average a human is responsible for 5t CO_2 per year and Strubell et al. (2019b) estimated that the training procedure of a big Transformer with neural architecture search emitted 284t of CO_2. Works (Lottick et al., 2019, Henderson et al. (2020)) have released online tools to benchmark their energy usage and initiatives such as the SustainNLP workshop have since taken up the goal of prioritizing computationally efficient hardware and algorithms. These findings are just some points one should keep in mind.

In the following chapters we will see how the multimodal architectures use these datasets and also how they perform on the given benchmarks.

3

Multimodal architectures

Authors: Luyang Chu, Karol Urbanczyk, Giacomo Loss, Max Schneider, Steffen Jauch-Walser

Supervisor: Christian Heumann

Multimodal learning refers to the process of learning representations from different types of input modalities, such as image data, text or speech. Due to methodological breakthroughs in the fields of Natural Language Processing (NLP) as well as Computer Vision (CV) in recent years, multimodal models have gained increasing attention as they are able to strengthen predictions and better emulate the way humans learn. This chapter focuses on discussing images and text as input data. The remainder of the chapter is structured as follows:

The first part "Image2Text" discusses how transformer-based architectures improve meaningful captioning for complex images using a new large scale, richly annotated dataset COCO (Lin et al., 2014c; Cornia et al., 2020). While looking at a photograph and describing it or parsing a complex scene and describing its context is not a difficult task for humans, it appears to be much more complex and challenging for computers. We start with focusing on images as input modalities. In 2014 Microsoft COCO was developed with a primary goal of advancing the state-of-the-art (SOTA) in object recognition by diving deeper into a broader question of scene understanding (Lin et al., 2014c). "COCO" in this case is the acronym for *Common Objects in Context*. It addresses three core problems in scene understanding: object detection (non-iconic views), segmentation, and captioning. While for tasks like machine translation and language understanding in NLP, transformer-based architecture are already widely used, the potential for applications in the multi-modal context has not been fully covered yet. With the help of the MS COCO dataset, the transformer-based architecture "Meshed-Memory Transformer for Image Captioning" (M^2) will be introduced, which was able to improve both image encoding and the language generation steps (Cornia et al., 2020). The performance of M^2 and other different fully-attentive models will be compared on the MS COCO dataset.

Next, in *Text2Image*, the idea of incorporating textual input in order to generate visual representations is described. Current advancements in this field have been made possible largely due to recent breakthroughs in NLP, which first allowed

for learning contextual representations of text. Transformer-like architectures are being used to encode the input into embedding vectors, which are later helpful in guiding the process of image generation. The chapter discusses the development of the field in chronological order, looking into details of the most recent milestones. Concepts such as generative adversarial networks (GAN), variational auto-encoders (VAE), VAE with vector quantization (VQ-VAE), diffusion, and autoregressive models are covered to provide the reader with a better understanding of the roots of the current research and where it might be heading. Some of the most outstanding outputs generated by state-of-the-art works are also presented in the chapter.

The third part, "Images supporting Language Models", deals with the integration of visual elements in pure textual language models. Distributional semantic models such as Word2Vec and BERT assume that the meaning of a given word or sentence can be understood by looking at how (in which context) and when the word or the sentence appear in the text corpus, namely from its "distribution" within the text. But this assumption has been historically questioned, because words and sentences must be grounded in other perceptual dimensions in order to understand their meaning (see for example the "symbol grounding problem"; Harnad, 1990). For these reasons, a broad range of models has been developed with the aim to improve pure language models, leveraging the addition of other perceptual dimensions, such as the visual one. This subchapter focuses in particular on the integration of visual elements (here: images) to support pure language models for various tasks at the word-/token-level as well as on the sentence-level. The starting point in this case is always a language model, into which visual representations (extracted often with the help of large pools of images rom data sets like MS COCO, see chapter "Img2Text" for further references) are to be "integrated". But how? There has been proposed a wide range of solutions: On one side of the spectrum, textual elements and visual ones are learned separately and then "combined" afterwards, whereas on the other side, the learning of textual and visual features takes place simultaneously/jointly.

For example, Silberer and Lapata (2014) implement a model where a one-to-one correspondence between textual and visual space is assumed. Text and visual representations are passed to two separate unimodal encoders and both outputs are then fed to a bimodal autoencoder. On the other side, Bordes et al. (2020) propose a "text objective function" whose parameters are shared with an additional "grounded objective function". The training of the latter takes place in what the authors called a "grounded space", which allows to avoid the one-to-one correspondence between textual and visual space. These are just introductory examples and between these two approaches there are many shades of gray (probably even more than fifty ..). These models exhibit in many instances better performance than pure language models, but they still struggle on some aspects, for example when they deal with abstract words and sentences.

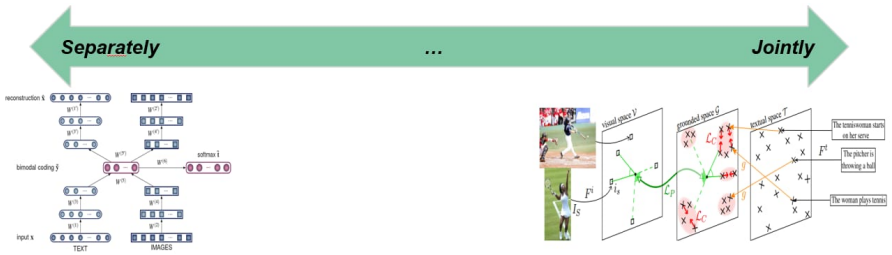


FIGURE 3.1: Left: Silberer and Lapata (2014) stack autoencoders to learn higher-level embeddings from textual and visual modalities, encoded as vectors of attributes. Right: Bordes et al. (2020) fuse textual and visual information in an intermediate space denoted as "grounded space"; the "grounding objective function" is not applied directly on sentence embeddings but trained on this intermediate space, on which sentence embeddings are projected.

Afterwards, in the subchapter on "Text supporting Image Models", approaches where natural language is used as additional supervision for CV models are described. Intuitively these models should be more powerful compared to models supervised solely by manually labeled data, simply because there is much more signal available in the training data.

One prominent example for this is the CLIP model (Radford et al., 2021a) with its new dataset WIT (WebImageText) comprising 400 million text-image pairs scraped from the internet. Similar to "Text2Image" the recent success stories in NLP have inspired most of the new approaches in this field. Most importantly pre-training methods, which directly learn from raw text (e.g. GPT-n, Generative Pre-trained Transformer; Brown et al., 2020). So, the acronym CLIP stands for _C_ontrastive _L_anguage-_I_mage _P_re-training here. A transformer-like architecture is used for jointly pre-training a text encoder and an image encoder. For this, the contrastive goal to correctly predict which natural language text pertains to which image inside a certain batch, is employed. Training this way turned out to be more efficient than to generate captions for images. This leads to a flexible model, which at test time uses the Learned text encoder as a "zero-shot" classifier on embeddings of the target dataset's classes. The model, for example, can perform optical character recognition, geo-location detection and action-recognition. Performance-wise CLIP can be competitive with task-specific supervised models, while never seeing an instance of the specific dataset before. This suggests an important step towards closing the "robustness gap", where machine learning models fail to meet the expectations set by their previous performance – especially on ImageNet test-sets – on new datasets.

Finally, the subchapter "Models for both modalities" discusses how text and image inputs can be incorporated into a single unifying framework in order to

get closer to a general self-supervised learning framework. There are two key advantages that make such an architecture particularly interesting. Similar to models mentioned in previous parts, devoid of human labelling, self-supervised models don't suffer from the same capacity constraints as regular supervised learning models. On top of that, while there have been notable advances in dealing with different modalities using single modality models, it is often unclear to which extend a model structure generalizes across different modalities. Rather than potentially learning modality-specific biases, a general multipurpose framework can help increase robustness while also simplifying the learner portfolio. In order to investigate different challenges and trends in vision-and-language modelling, this section takes a closer look at three different models, namely data2vec (Baevski et al. (2022)), VilBert (Lu et al. (2019b)) and Flamingo (Alayrac et al. (2022)) Data2vec is a new multimodal self-supervised learning model which uses a single framework to process either speech, natural language or visual information. This is in contrast to earlier models which used different algorithms for different modalities. The core idea of data2vec, developed by MetaAI, is to predict latent representations of the full input data based on a masked view of the input in a self-distillation setup using a standard transformer architecture. (Baevski et al. (2022)) As a result, the main improvement is in the framework itself, not the underlying architectures themselves. For example, the transformer architecture being used follows Vaswani et al. (2017b). Through their parallelizability, transformers have several advantages over RNNs/CNNs particularly when large amounts of data are being used, making them the de-facto standard approach in vision-language modelling. (Dosovitskiy et al. (2020a)) VilBert is an earlier model that in contrast to data2vec can handle cross-modality tasks. Finally, Flamingo is a modern few shot learning model which features 80B parameters - significantly more than the other two models. Through a large language model incorporated in its architecture, it has great text generating capabilities to tackle open-ended tasks. It also poses the question how to efficiently train increasingly large models and shows the effectiveness of using perceiver architectures (Jaegle et al. (2021a)) to encode inputs from different modalities as well as how to leverage communication between pretrained and frozen models.

3.1 Image2Text

Author: Luyang Chu

Supervisor: Christian Heumann

Image captioning refers to the task of producing descriptive text for given images. It has stimulated interest in both natural language processing and computer vision research in recent years. Image captioning is a key task that

requires a semantic comprehension of images as well as the capacity to generate accurate and precise description sentences.

3.1.1 Microsoft COCO: Common Objects in Context

The uderstanding of visual scenes plays an important role in computer vision (CV) research. It includes many tasks, such as image classification, object detection, object localization and semantic scene labeling. Through the CV research history, high-quality image datasets have played a critical role. They are not only essential for training and evaluating new algorithms, but also lead the research to new challenging directions (Lin et al., 2014c). In the early years, researchers developed Datasets (Deng et al., 2009),(Xiao et al., 2010),(Everingham et al., 2010) which enabled the direct comparison of hundreds of image recognition algorithms, which led to an early evolution in object recognition. In the more recent past, ImageNet (Deng et al., 2009), which contains millions of images, has enabled breakthroughs in both object classification and detection research using new deep learning algorithms.

With the goal of advancing the state-of-the-art in object recognition, especially scene understanding, a new large scale data called "Microsoft Common Objects in Context" (MS COCO) was published in 2014. MS COCO focuses on three core problems in scene understanding: detecting non-iconic views, detecting the semantic relationships between objects and determining the precise localization of objects (Lin et al., 2014c).

The MS COCO data set contains 91 common object categories with a total of 328,000 images as well as 2,500,000 instance labels. The authors claim, that all of these images could be recognized by a 4 year old child. 82 of the categories include more than 5000 labeled instances. These labeled instances wmay support the detection of relationships between objects in MS COCO. In order to provide precise localization of object instances, only "Thing" categories like e.g. car, table, or dog were included. Objects which do not have clear boundaries like e.g. sky, sea, or grass, were not included. In current object recognition research, algorithms perform well on images with iconic views. Images with iconic view are defined as containing the one single object category of interest in the center of the image. To accomplish the goal of detecting the contextual relationships between objects, more complex images with multiple objects or natural images, coming from our daily life, are also gathered for the data set.

In addition to MS COCO, researchers have been working on the development of new large databases. In recent years many new large databases like ImageNet, PASCAL VOC (Everingham et al., 2010) and SUN (Xiao et al., 2010) have been developed in the field of computer vision. Each of this dataset has its on specific focus.

Datasets for object recognition can be roughly split into three groups: object classification, object detection and semantic scene labeling.

Object classification requires binary labels to indicate whether objects are present in an image, ImageNet (Deng et al., 2009) is clearly distinguishable from other datasets in terms of the data set size. ImageNet contains 22k categories with 500-1000 images each.In comparison to other data sets, the ImageNet data set contains thus over 14 million labeled images with both entity-level and fine-grained categories by using the WordNet hierarchy and has enabled significant advances in image classification.

Detecting an object includes two steps: first is to ensure that an object from a specified class is present, the second step is to localize the object in the image with a given bounding box. This can be implemented to solve tasks like face detection or pedestrians detection. The PASCAL VOC (Everingham et al., 2010) data set can be used to help with the detection of basic object categories. With 20 object categories and over 11,000 images, PASCAL VOC contains over 27,000 labeled object instances by additionally using bounding boxes. Almost 7,000 object instances from them come with detailed segmentations (Lin et al., 2014c).

Labeling semantic objects in a scene requires that each pixel of an image is labeled with respect to belonging to a category, such as sky, chair, etc., but individual instances of objects do not need to be segmented (Lin et al., 2014c). Some objects like sky, grass, street can also be defined and labeled in this way. The SUN data set (Xiao et al., 2010) combines many of the properties of both object detection and semantic scene labeling data sets for the task of scene understanding, it contains 908 scene categories from the WordNet dictionary (Fellbaum, 2000) with segmented objects. The 3,819 object categories split them to object detection datasets (person, chair) and to semantic scene labeling (wall, sky, floor) (Lin et al., 2014c).

3.1.1.1 Image Collection and Annotation for MS COCO

MS COCO is a large-scale richly annotated data set, the progress of building consisted of two phases: data collection and image annotation.

In order to select representative object categories for images in MS COCO, researchers collected several categories from different existing data sets like PASCAL VOC (Everingham et al., 2010) and other sources. All these object categories could, according to the authors, be recognized by children between 4 to 8. The quality of the object categories was ensured by co-authors. Co-authors rated the categories on a scale from 1 to 5 depending on their common occurrence, practical applicability and diversity from other categories (Lin et al., 2014c). The final number of categories on their list was 91. All the categories from PASCAL VOC are included in MS COCO.

With the help of representative object categories, the authors of MS COCO

wanted to collect a data set in which a majority of the included images are non-iconic. All included images can be roughly divided into three types according to Fig. 3.2: iconic-object images, iconic-scene images and non-iconic images (Lin et al., 2014c).

Fig. 2: Example of (a) iconic object images, (b) iconic scene images, and (c) non-iconic images.

FIGURE 3.2: Type of images in the data set (Lin et al., 2014c).

Images are collected through two strategies: firstly images from Flickr, a platform for photos uploaded by amateur photographers, with their keywords are collected. Secondly, researchers searched for pairwise combinations of object categories like "dog + car" to gather more non-iconic images and images with rich contextual relationships (Lin et al., 2014c).

Due to the scale of the dataset and the high cost of the annotation process, the design of a high quality annotation pipeline with efficient cost depicted a difficult task. The annotation pipeline in Fig. 3.3 for MS COCO was split into three primary tasks: 1. category labeling, 2.instance spotting, and 3. instance segmentation (Lin et al., 2014c).

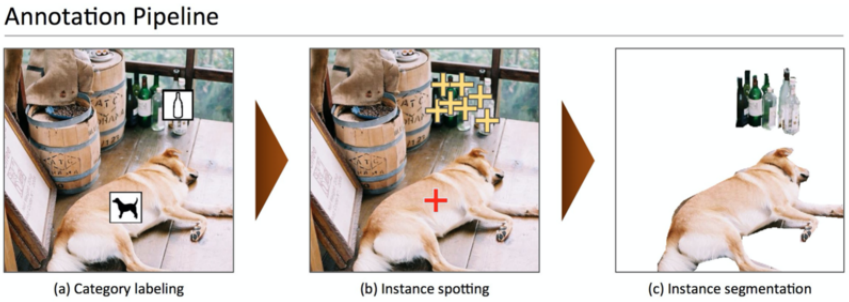


FIGURE 3.3: Annotation pipeline for MS COCO (Lin et al., 2014c).

As we can see in the Fig 3.3, object categories in each image were determined in the first step. Due to the large number of data sets and categories, they used a hierarchical approach instead of doing binary classification for each category. All the 91 categories were grouped into 11 super-categories. The

annotator did then examine for each single instance whether it belongs to one of the given super-categories. Workers only had to label one instance for each of the super-categories with a category's icon (Lin et al., 2014c). For each image, eight workers were asked to label it. This hierarchical approach helped to reduce the time for labeling. However, the first phase still took 20k worker hours to be completed.

In the next step, all instances of the object categories in an image were labeled, at most 10 instances of a given category per image were labeled by each worker. In both the instance spotting and the instance segmentation steps, the location of the instance found by a worker in the previous stage could be seen by the current worker. Each image was labeled again by eight workers summing up to a total of 10k worker hours.

In the final segmenting stage, each object instance was segmented, the segmentation for other instances and the specification of the object instance by a worker in the previous stage were again shown to the worker. Segmenting 2.5 million object instances was an extremely time consuming task which required over 22 worker hours per 1,000 segmentations. To minimize cost and improve the quality of segmentation, all workers were required to complete a training task for each object category. In order to ensure a better quality, an explicit verification step on each segmented instance was performed as well.

3.1.1.2 Comparison with other data sets

In recent years, researchers have developed several pre-training data sets and benchmarks which helped the developemnt of algorithms for CV. Each of these data sets varies significantly in size, number of categories and types of images. In the previos part, we also introduced the different research focus of some data sets like e.g. ImageNet (Deng et al., 2009), PASCAL VOC (Everingham et al., 2010) and SUN (Xiao et al., 2010). ImageNet, containing millions of images, has enabled major breakthroughs in both object classification and detection research using a new class of deep learning algorithms. It was created with the intention to capture a large number of object categories, many of which are fine-grained. SUN focuses on labeling scene types and the objects that commonly occur in them. Finally, PASCAL VOC's primary application is in object detection in natural images. MS COCO is designed for the detection and segmentation of objects occurring in their natural context (Lin et al., 2014c).

With the help of Fig. 3.4, one can compare MS COCO to ImageNet, PASCAL VOC and SUN with respect to different aspects (Lin et al., 2014c).

The number of instances per category for all 91 categories in MS COCO and PASCAL VOC is shown in subfigure 3.4 (a). Compared to PASCAL VOC, MS COCO has both more categories and (on average) more instances per category. The number of object categories and the number of instances per

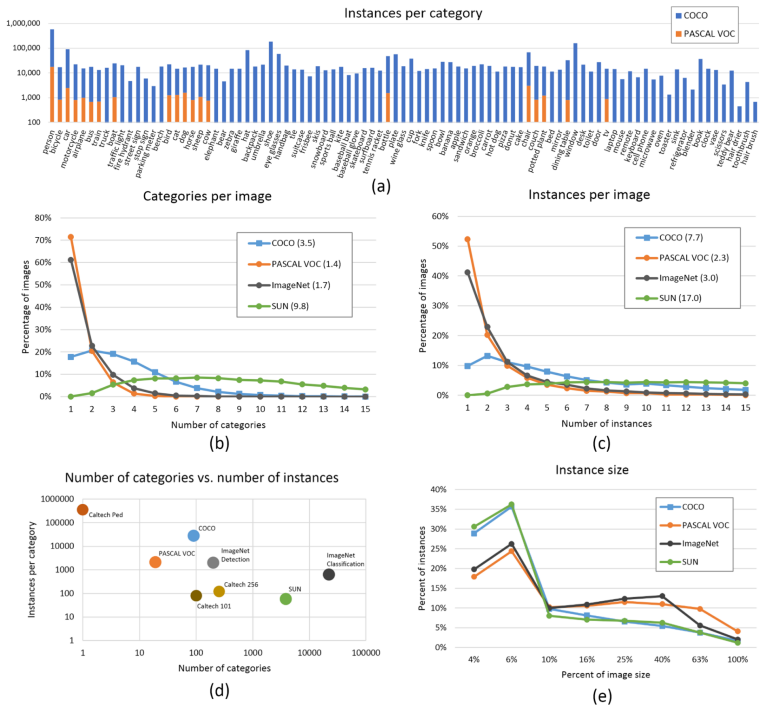


FIGURE 3.4: Comparison MS COCO with PASCAL VOC, SUN and ImageNet (Lin et al., 2014c).

category for all the datasets is shown in subfigure 3.4 (d). MS COCO has fewer categories than ImageNet and SUN, but it has the highest average number of instances per category among all the data sets, which from the perspective of authors might be useful for learning complex models capable of precise localization (Lin et al., 2014c). Subfigures 3.4 (b) and (c) show the number of annotated categories and annotated instances per image for MS COCO, ImageNet, PASCAL VOC and SUN (average number of categories and instances are shown in parentheses). On average MS COCO contains 3.5 categories and 7.7 instances per image. ImageNet and PASCAL VOC both have on average less than 2 categories and 3 instances per image. The SUN data set has the most contextual information, on average 9.8 categories and 17 instances per image. Subfigure 3.4 (e) depicts the distribution of instance sizes for the MS COCO, ImageNet Detection, PASCAL VOC and SUN data set.

3.1.1.3 Discussion

MS COCO is a large scale data set for detecting and segmenting objects found in everyday life, with the aim of improving the state-of-the-art in object

recognition and scene understanding. It focuses on non-iconic images of objects in natural environments and contains rich contextual information with many objects present per image. MS COCO is one of the typically used vision data sets, which are labor intensive and costly to create. With the vast cost and over 70,000 worker hours, 2.5 Mio instances were annotated to drive the advancement of object detection and segmentation algorithms. MS COCO is still a good benchmark for the field of CV (Lin et al., 2014c). The MS COCO Team also shows directions for future. For example "stuff" label like "sky", "grass", and "street", etc, may also be included in the dataset since "stuff" categories provide significant contextual information for the object detection.

3.1.2 Models for Image captioning

The image captioning task is generally to describe the visual content of an image in natural language, so it requires an algorithm to understand and model the relationships between visual and textual elements, and to generate a sequence of output words (Cornia et al., 2020). In the last few years, collections of methods have been proposed for image captioning. Earlier approaches were based on generations of simple templates, which contained the output produced from the object detector or attribute predictor (Socher and Fei-fei, 2010), (Yao et al., 2010). With the sequential nature of language, most research on image captioning has focused on deep learning techniques, using especially Recurrent Neural Network models (RNNs) (Vinyals et al., 2015), (Karpathy and Fei-Fei, 2014) or one of their special variants (e.g. LSTMs). Mostly, RNNs are used for sequence generation as languages models, while visual information is encoded in the output of a CNN. With the aim of modelling the relationships between image regions and words, graph convolution neural networks in the image encoding phase (Yao et al., 2018a) or single-layer attention mechanisms (Xu et al., 2015) on the image encoding side have been proposed to incorporate more semantic and spatial relationships between objects. RNN-based models are widely adopted, however, the model has its limitation on representation power and due to its sequential nature (Cornia et al., 2020). Recently, new fully-attentive models, in which the use of self-attention has replaced the recurrence, have been proposed. New approaches apply the Transformer architecture (Vaswani et al., 2017d) and BERT (Devlin et al., 2019) models to solve image captioning tasks. The transformer consists of an encoder with a stack of self-attention and feed-forward layers, and a decoder which uses (masked) self-attention on words and cross-attention over the output of the last encoder layer (Cornia et al., 2020). In some other transformer-based approaches, a transformer-like encoder was paired with an LSTM decoder, while the aforementioned approaches have exploited the original transformer architecture. Others (Herdade et al., 2019) proposed a transformer architecture for image captioning with the focus on geometric relations between input objects at the same time. Specifically, additional geometric weights between object pairs, which is used to scale attention weights, are computed. Similarly,

an extension of the attention operator, in which the final attended information is weighted by a gate guided by the context, was introduced at a similar time (Huang et al., 2019).

3.1.3 Meshed-Memory Transformer for Image Captioning (M^2)

Transformer-based architectures have been widely implemented in sequence modeling tasks like machine translation and language understanding. However, their applicability for multi-modal tasks like image captioning has still been largely under-explored (Cornia et al., 2020).

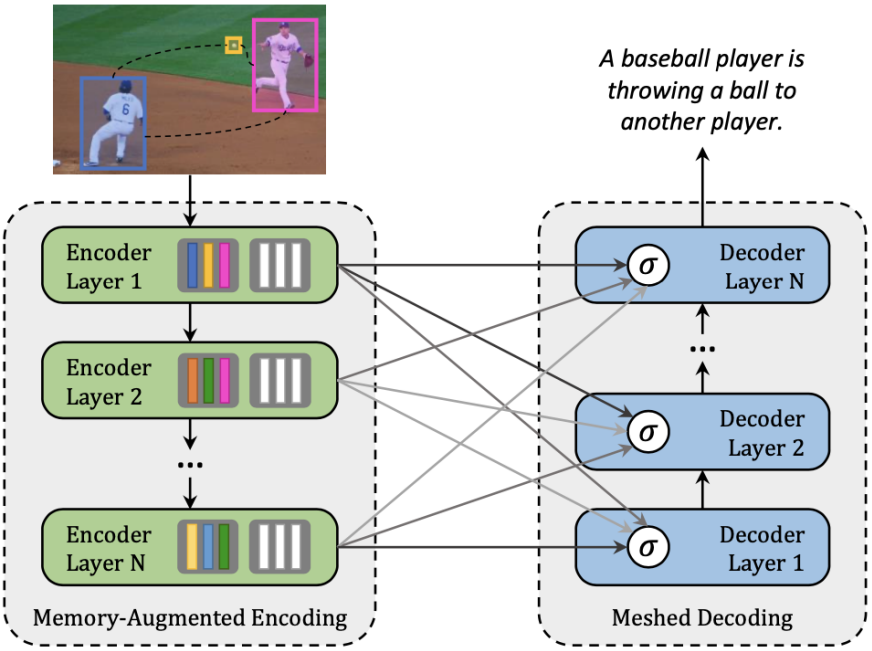


FIGURE 3.5: M^2 Transformer (Cornia et al., 2020).

A novel fully-attentive approach called Meshed-Memory Transformer for Image Captioning (M^2) was proposed in 2020 (Cornia et al., 2020) with the aim of improving the design of both the image encoder and the language decoder. Compared to all previous image captioning models, M^2 (see Fig. 3.5 has two new novelties: The encoder encodes a multi-level representation of the relationships between image regions with respect to low-level and high-level relations, and a-priori knowledge can be learned and modeled by using persistent

memory vectors. The multi-layer architecture exploits both low- and high-level visual relationships through a learned gating mechanism, which computes the weight at each level, therefore, a mesh-like connectivity between encoder and decoder layers is created for the sentence generation process (Cornia et al., 2020).

3.1.3.1 M^2 Transformer Architecture

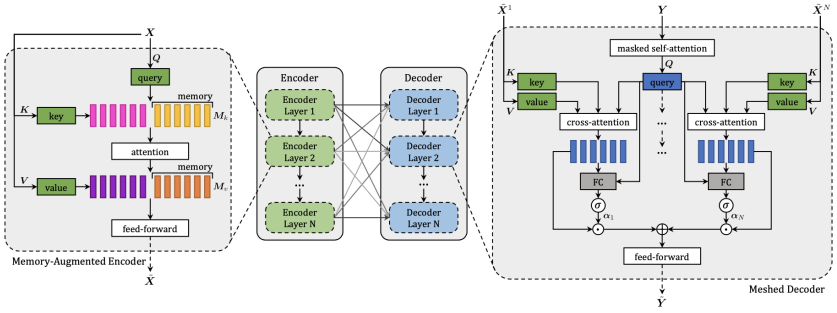


FIGURE 3.6: M^2 Transformer Architecture (Cornia et al., 2020).

Fig. 3.6 shows the detailed architecture of M^2 Transformer. It can be divided into the encoder (left) module and the decoder (right) module, both modules with multiple layers. Given the input image region X, the image is passed through the attention and feed forward layers. The relationship between image regions with a-priori knowledge will be encoded in each encoding layer, the output of each encoding layers will be read by decoding layers to generate the caption for image word by word (Cornia et al., 2020).

All interactions between word and image-level features of the input image X are modeled by using scaled dot-product attention. Attention operates on vectors of queries q, keys k and values n, and takes a weighted sum of the value vectors according to a similarity distribution between query and key vectors. Attention can be defined as follows (Cornia et al., 2020):

$$Attention(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d}})V \tag{3.1}$$

where Q is a matrix of n_q query vectors, K and V both contain n_k keys and values, all the vectors has the same dimensionality, and d is a scaling factor.

3.1.3.1.1 Memory-Augmented Encoder

For the given image region X, attention can be used to obtain a permutation invariant encoding of X through the self-attention operations, the operator from the Transformer can be defined as follows (Cornia et al., 2020):

$$S(X) = Attention(W_q X, W_k X, W_v X) \tag{3.2}$$

In this case, queries, keys, and values are linear projections of the input features, and W_q, W_k, W_v are their learnable weights, they depend solely on the pairwise similarities between linear projections of the input set X. The self-attention operator encodes the pairwise relationships inside the input. But self-attention also has its limitation: a prior knowledge on relationships between image regions can not be modelled. To overcome the limitation, the authors introduce a **Memory-Augmented Attention** operator by extending the keys and values with additional prior information, which does not depend on image region X. The additional keys and values are initialized as plain learnable vectors which can be directly updated via SGD. The operator can be defined as follows (Cornia et al., 2020):

$$\begin{aligned} M_{mem}(X) &= Attention(W_q X, K, V) \\ K &= [W_k X, M_k] \\ V &= [W_v X, M_v] \end{aligned} \tag{3.3}$$

M_k and M_v are learnable matrices, with n_m rows, $[\,\cdot\,,\,\cdot\,]$ indicates concatenation. The additional keys and value could help to retrieve a priori knowledge from input while keeping the quries unchanged (Cornia et al., 2020).

For the **Encoding Layer**, a memory-augmented operator d is injected into a transformer-like layer, the output is fed into a position-wise feed-forward layer (Cornia et al., 2020):

$$F(X)_i = U\sigma(VX_i + b) + c; \tag{3.4}$$

X_i indicates the i-th vector of the input set, and $F(X)_i$ the i-th vector of the output. Also, $\sigma(\mathring{u})$ is the ReLU activation function, V and U are learnable weight matrices, b and c are bias terms (Cornia et al., 2020).

Each component will be complemented by a residual connection and the layer norm operation. The complete definition of an encoding layer can be finally written as (Cornia et al., 2020):

$$\begin{aligned} Z &= AddNorm(M_{mem}(X)) \\ \tilde{X} &= AddNorm(F(Z)) \end{aligned} \tag{3.5}$$

Finally the **Full Encoder** has multiple encoder layers in a sequential fashion, therefore the i-th layer uses the output set computed by layer $i-1$, higher encoding layers can exploit and refine relationships identified by previous layers, n encoding layers will produce the output $\tilde{X} = (\tilde{X}^1 \dots \tilde{X}^n)$ (Cornia et al., 2020).

3.1.3.1.2 Meshed Decoder

The decoder depends on both previously generated words and image region encodings. **Meshed Cross-Attention** can take advantage of all the encoder layers to generate captions for the image. On the right side of the Fig. 3.6 the structure of the meshed decoder is shown. The input sequence vector Y and the outputs from all encoder layers $\tilde{X}$ are connected by the meshed attention operator gated through cross-attention. The meshed attention operator can is formally defined as (Cornia et al., 2020):

$$M_{mesh}(\tilde{X}, Y) = \sum_{i=1}^{N} \alpha_i C(\tilde{X}^i, Y) \tag{3.6}$$

$C(\mathring{u}, \mathring{u})$ stands for the encoder-decoder cross-attention, it is defined with queries from decoder, while the keys and values come from the encoder (Cornia et al., 2020).

$$C(\tilde{X}^i, Y) = Attention(W_q Y, W_k \tilde{X}^i, W_v \tilde{X}^i) \tag{3.7}$$

α_i is a matrix of weights of the same size as the cross-attention results, α_i models both single contribution of each encoder layer and the relative importance between different layers (Cornia et al., 2020).

$$\alpha_i = \sigma(W_i[Y, C(\tilde{X}^i, Y)] + b_i) \tag{3.8}$$

The $[\,\cdot\,,\cdot\,]$ indicates concatenation and $\sigma(\mathring{u})$ is the sigmoid activation function here, W_i is a weight matrix, and b_i is a learnable bias vector (Cornia et al., 2020).

In decoder layers the prediction of a word should only depend on the previously generated word, so the decoder layer comprises a masked self-attention operation, which means that the operator can only make connections between queries derived from the t-th element of its input sequence Y with keys and values from left sub-sequence, i.e. $Y_{\leq t}$.

Simlilar as the encoder layers, the decoder layers also contain a position-wise feed-forward layer, so the decoder layer can be finally defined as (Cornia et al., 2020):

$$\begin{aligned} Z &= AddNorm(M_{mesh}(X, AddNorm(S_{mask}(Y)))) \\ \tilde{Y} &= AddNorm(F(Z)), \end{aligned} \tag{3.9}$$

where S_{mask} indicates a masked self-attention over time (Cornia et al., 2020). The full decoder with multiple decoder layers takes the input word vectors as well as the t-th element (and all elements prior to it) of its output sequence

to make the prediction for the word at $t+1$, conditioned on $Y_{\leq t}$. Finally the decoder takes a linear projection and a softmax operation, which can be seen as a probability distribution over all words in the vocabulary (Cornia et al., 2020).

3.1.3.1.3 Comparison with other models on the MS COCO data sets

The M^2 Transformer was evaluated on MS COCO, which is still one of the most commonly used test data set for image captioning. Instead of using the original MS COCO dat set, Cornia et al. (2020) follow the split of MS COCO provided by Karpathy and Fei-Fei (2014). Karpathy uses 5000 images for validation, 5000 images for testing and the rest for training.

For model evaluation and comparison, standard metrics for evaluating generated sequences, like BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ROUGE (Lin, 2004), CIDEr (Vedantam et al., 2015), and SPICE (Anderson et al., 2016), which have been introduced in the second chapter, are used.

	B-1	B-4	M	R	C	S
Transformer (w/ 6 layers as in [39])	79.1	36.2	27.7	56.9	121.8	20.9
Transformer (w/ 3 layers)	79.6	36.5	27.8	57.0	123.6	21.1
Transformer (w/ AoA [14])	80.3	38.8	29.0	58.4	129.1	**22.7**
$\mathcal{M}^2$ Transformer$^{\text{1-to-1}}$ (w/o mem.)	80.5	38.2	28.9	58.2	128.4	22.2
$\mathcal{M}^2$ Transformer$^{\text{1-to-1}}$	80.3	38.2	28.9	58.2	129.2	22.5
$\mathcal{M}^2$ Transformer (w/o mem.)	80.4	38.3	29.0	58.2	129.4	22.6
$\mathcal{M}^2$ Transformer (w/ softmax)	80.3	38.4	29.1	58.3	130.3	22.5
$\mathcal{M}^2$ **Transformer**	**80.8**	**39.1**	**29.2**	**58.6**	**131.2**	22.6

FIGURE 3.7: Comparison of M^2 with Transformer-based alternatives (Cornia et al., 2020)

The transformer architecture in its original configuration with six layers has been applied for captioning, researchers speculated that specific architectures might be required for captioning, so variations of the original transformer are compared with M^2 Transformer. Other variations are a transformer with three layers and the "Attention on Attention" (AoA) approach (Huang et al., 2019) to the attentive layers, both in the encoder and in the decoder (Cornia et al., 2020). The second part intends to evaluate the importance of the meshed connections between encoder and decoder layers. M^2 Transformer (1 to 1) is a reduced version of the original M^2 Transformer, in which one encoder layer is connected to only corresponding decoder layer instead of being

connected to all the decoder layers. As one can see from the Fig. 3.7, the original Transformer has a 121.8 CIDEr score, which is lower than the reduced version of M^2 Transformer, showing an improvement to 129.2 CIDEr. With respect to meshed connectivity, which helps to exploit relationships encoded at all layers and weights them with a sigmoid gating, one can observe a further improvement in CIDEr from 129.2 to 131.2. Also the role of memory vectors and the softmax gating schema for M^2 Transformer are also included in the table. Eliminating the memory vector leads to a reduction of the performance by nearly 1 point in CIDEr in both the reduced M^2 Transformer and the original M^2 Transformer (Cornia et al., 2020).

	B-1	B-4	M	R	C	S
SCST [33]	-	34.2	26.7	55.7	114.0	-
Up-Down [4]	79.8	36.3	27.7	56.9	120.1	21.4
RFNet [15]	79.1	36.5	27.7	57.3	121.9	21.2
Up-Down+HIP [49]	-	38.2	28.4	58.3	127.2	21.9
GCN-LSTM [48]	80.5	38.2	28.5	58.3	127.6	22.0
SGAE [46]	**80.8**	38.4	28.4	58.6	127.8	22.1
ORT [13]	80.5	38.6	28.7	58.4	128.3	**22.6**
AoANet [14]	80.2	38.9	**29.2**	**58.8**	129.8	22.4
$\mathcal{M}^2$ **Transformer**	**80.8**	**39.1**	**29.2**	58.6	**131.2**	**22.6**

FIGURE 3.8: Comparison with the state-of-the-art on the "Karpathy" test split, in single-model setting (Cornia et al., 2020).

Fig 3.8 compares the performance of M^2 Transformer with several recently proposed models for image captioning. SCST (Rennie et al., 2017) and Up-Down (Anderson et al., 2018), use attention over the grid of features and attention over regions. RFNet (**?**) uses a recurrent fusion network to merge different CNN features; GCN-LSTM (Yao et al., 2018b) uses a Graph CNN to exploit pairwise relationships between image regions; SGAE (Yang et al., 2019) uses scene graphs instead ofauto-encoding. The original AoA-Net (Huang et al., 2019) approach uses attention on attention for encoding image regions and an LSTM language model. Finally, the ORT (Herdade et al., 2019) uses a plain transformer and weights attention scores in the region encoder with pairwise distances between detections (Cornia et al., 2020).

In Fig. 3.8, the M^2 Transformer exceeds all other models on BLEU-4, METEOR, and CIDEr. The performance of the M^2 Transformer was very close and competitive with SGAE on BLEU-1 and with ORT with respect to SPICE.

FIGURE 3.9: Examples of captions generated by M^2 Transformer and the original Transformer model, as well as the corresponding ground-truths (Cornia et al., 2020).

Fig. 3.9 shows some examples of captions generated by M^2 Transformer and the original transformer model, as well as the corresponding ground-truths. According to the selected examples of captions, M^2 Transformer shows the ability to generate more accurate descriptions of the images, and the approach could detect the more detailed relationships between image regions (Cornia et al., 2020).

The M^2 Transformer is a new transformer-based architecture for image captioning. It improves the image encoding by learning a multi-level representation of the relationships between image regions while exploiting a priori knowledge from each encoder layer, and uses a mesh-like connectivity at decoding stage to exploit low- and high-level features at the language generation steps. The results of model evaluation with MS COCO shows that the performance of the M^2 Transformer approach surpasses most of the recent approaches and achieves a new state of the art on MS COCO (Cornia et al., 2020).

3.2 Text2Image

Author: Karol Urbańczyk

Supervisor: Jann Goschenhofer

Have you ever wondered what a painting artist could paint for you if you ordered *a high-quality oil painting of a psychedelic hamster dragon*? Probably not. Nevertheless, one of the answers could be:

FIGURE 3.10: Hamster dragon

The catch is that there is no human artist. The above picture comes from a 3.5-billion parameter model called GLIDE by OpenAI (Nichol et al., 2021b). Every single value of every pixel was generated from a distribution that the model had to learn in the first place. Before generating the image, GLIDE abstracted the concepts of 'hamster' and 'dragon' from looking at millions of training images. Only then, it was able to create and combine them successfully into a meaningful visual representation. Welcome to the world of current text-to-image modelling!

The cross-modal field of text-to-image models has developed significantly over recent years. What was considered unimaginable only a few years ago, today constitutes a new benchmark for researchers. New breakthroughs are being published every couple of months. Following these, possible business use cases are emerging, which attracts investment from the greatest players in AI research. However, a further trend of closed-source models is continuing and the text-to-image field is probably one the most obvious ones where it can be noticed. We might need to get used to the fact that the greatest capabilities will soon be monopolized by few companies.

At the same time, the general public is becoming aware of the field itself and the disruption potential it brings. Crucial questions are already emerging.

What constitutes art? What does the concept of being an author mean? The result of a generative model is in a sense a combination, or variation, of the abstracts it has seen in the past. But the same stands for a human author. Therefore, is a discussion about the prejudices and biases needed? Answers to all of these will require refinement through an extensive discussion. The last section of this chapter will try to highlight the most important factors that will need to be considered.

However, the primary intention of this chapter is to present the reader with a perspective on how the field was developing chronologically. Starting with the introduction of GANs, through the first cross-domain models, and ending with state-of-the-art achievements (as of September 2022), it will also try to grasp the most important concepts without being afraid of making technical deep dives.

The author is aware that since the rapid development pace makes it nearly impossible for this section to stay up-to-date, it might very soon not be fully covering the field. However, it must be stressed that the cutting-edge capabilities of the recent models tend to come from the scale and software engineering tricks. Therefore, focusing on the core concepts should hopefully gives this chapter a universal character, at least for some time. This design choice also explains why many important works did not make it to this publication. Just to name a few of them as honorable mentions: GAWWN (Reed et al., 2016a), MirrorGAN (Qiao et al., 2019), or most recent ones: LAFITE (Zhou et al., 2021), Make-a-Scene (Gafni et al., 2022) or CogView (Ding et al., 2021). In one way or another, all of them pushed the research frontier one step further. Therefore, it needs to be clearly stated - the final selection of this chapter's content is a purely subjective decision of the author.

3.2.1 Seeking objectivity

Before diving into particular models, we introduce objective evaluation procedures that help assess the performance of consecutive works in comparison to their predecessors. Unfortunately, objectivity in comparing generative models is very hard to capture since there is no straight way to draw deterministic conclusions about the model's performance (Theis et al., 2015). However, multiple quantitative and qualitative techniques have been developed to make up for it. Unfortunately, there is no general consensus as to which measures should be used. An extensive comparison has been performed by Borji (2018). A few of the most widely used ones in current research are presented below.

Inception Score (IS)

Introduced by Salimans et al. (2016), Inception Score (IS) uses the Inception Net (Szegedy et al., 2015) trained on ImageNet data to classify the fake images generated by the assessed model. Then, it measures the average KL diver-

gence between the marginal label distribution $p(y)$ and the label distribution conditioned on the generated samples $p(y|x)$.

$$exp(\mathop{\mathbb{E}}_{x}[KL(p(y|x)||p(y))])$$

$p(y)$ is desired to have high diversity (entropy), in other words: images from the generative model should represent a wide variety of classes. On the other hand, $p(y|x)$ is desired to have low diversity, meaning that images should represent meaningful concepts. If a range of cat images is being generated, they all should be confidently classified by Inception Net as cats. The intention behind IS is that a generative model with a higher distance (KL divergence in this case) between these distributions should have a better score. IS is considered a metric that correlates well with human judgment, hence its popularity.

Fréchet Inception Distance (FID)

A metric that is generally considered to improve upon Inception Score is the Fréchet Inception Distance (FID). Heusel et al. (2017) argue that the main drawback of IS is that it is not considering the real data at all. Therefore, FID again uses Inception Net, however this time it embeds the images (both fake and real samples) into feature space, stopping at a specific layer. In other words, some of the ultimate layers of the network are being discarded. Feature vectors are then assumed to follow a Gaussian distribution and the Fréchet distance is calculated between real and generated data distributions:

$$d^2((m, C), (m_w, C_w)) = ||m - m_w||_2^2 + Tr(C + C_w - 2(CC_w)^{1/2})$$

where (m, C) and (m_w, C_w) represent mean and covariance of generated and real data Gaussians respectively. Obviously, low FID levels are desired.

FID is considered to be consistent with human judgement and sensitive to image distortions, which are both desired properties. Figure 3.11 shows how FID increases (worsens) for different types of noise being added to images.

Precision / Recall

Precision and recall are one of the most widely used metrics in many Machine Learning problem formulations. However, their classic definition cannot be applied to generative models due to the lack of objective labels. Sajjadi et al. (2018) came up with a novel definition of these metrics calculated directly from distributions, which was further improved by Kynkäänniemi et al. (2019). The argument behind the need for such an approach is that metrics such as IS or FID provide only a one-dimensional view of the model's performance, ignoring the trade-off between precision and recall. A decent FID result might very well mean high recall (large variation, i.e. wide range of data represented by the model), high precision (realistic images), or anything in between.

Let P_r denote the probability distribution of the real data, and P_g be the

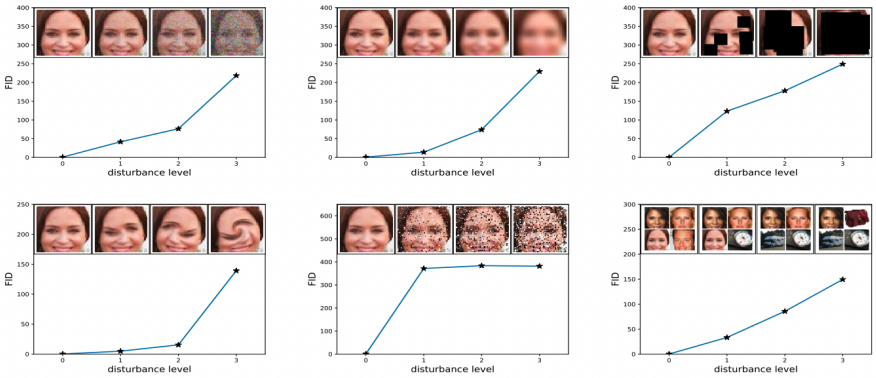


FIGURE 3.11: FID is evaluated for different noise types. From upper left to lower right: Gaussian noise, Gaussian blur, implanted black rectangles, swirled images, salt and pepper, CelebA dataset contaminated by ImageNet images. Figure from Heusel et al. (2017).

distribution of the generated data. In short, recall measures to which extend P_r can be generated from P_g, while precision is trying to grasp how many generated images fall within P_r.

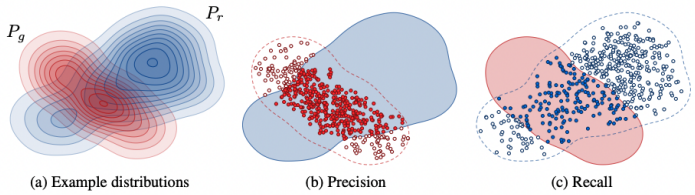


FIGURE 3.12: Definition of precision and recall for distributions. Figure from Kynkäänniemi et al. (2019).

See Kynkäänniemi et al. (2019) for a more thorough explanation.

CLIP score

CLIP is a model from OpenAI [CLIP2021] which is explained in detail in the chapter about *text-supporting computer vision models*. In principle, CLIP is capable of assessing the semantic similarity between the text caption and the image. Following this rationale, the CLIP score can be used as metric and is defined as:

$$\mathbb{E}[s(f(image) * g(caption))]$$

where the expectation is taken over the batch of generated images and s is the CLIP logit scale (Nichol et al., 2021b).

Human evaluations

It is common that researchers report also qualitative measures. Many potential applications of the models are focused on deceiving the human spectator, which motivates reporting of metrics that are based on human evaluation. The general concept of these evaluation is to test for:

- photorealism
- caption similarity (image-text alignment)

Usually, a set of images is presented to a human, whose task is to assess their quality with respect to the two above-mentioned criteria.

3.2.2 Generative Adversarial Networks

The appearance of Generative Adversarial Networks (GAN) was a major milestone in the development of generative models. Introduced by Goodfellow et al. (2014c), the idea of GANs presented a novel architecture and training regime, which corresponds to a minimax two-player game between a Generator and a Discriminator (hence the word *adversarial*).

GANs can be considered as an initial enabler for the field of text-to-image models and for a long time, GAN-like models were achieving state-of-the-art results, hence the presentation of their core concepts in this chapter

3.2.2.1 Vanilla GAN for Image Generation

In a vanilla GAN, the Generator model (G) and Discriminator model (D) are optimized together in a minimax game, where G aims at generating a sample so convincing, that D will not be able to distinguish whether it comes from a real or generated image distribution. On the other hand, D is being trained to discriminate between the two. Originally, a multilayer perceptron was proposed as a model architecture for both D and G, although in theory any differentiable function could be used.

More formally, let p_z denote the prior distribution defined on the input noise vector z. Then, the generator $G(z)$ represents a function that is mapping this noisy random input to the generated image x. The discriminator $D(x)$ outputs a probability that x comes from the real data rather than generator's distribution p_g. In this framework, D shall maximize the probability of guessing the correct label of both real and fake data. G is trained to minimize $log(1 - D(G(z)))$. Now, such representation corresponds to the following value function (optimal solution):

$$\min_G \min_D V(D,G) = \mathop{\mathbb{E}}_{x \sim p_{data}(x)}[log(D(x))] + \mathop{\mathbb{E}}_{z \sim p_z(z)}[log(1 - D(G(z)))]$$

Figure 3.13 depicts this process in a visual way.

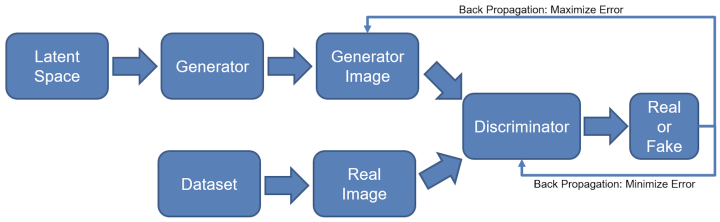


FIGURE 3.13: GAN framework as proposed in Goodfellow et al. (2014c).

Some of the generated samples that had been achieved with this architecture already in 2014 can be seen in Figure 3.14.

FIGURE 3.14: Samples from generators trained on different datasets: a) MNIST b) TFD, c) CIFAR-10 (MLP used for G and D) d) CIFAR-10 (CNN used). Highlighted columns show the nearest real example of the neighbouring sample. Figure from Goodfellow et al. (2014c).

3.2.2.2 Conditioning on Text

So far, only image generation has been covered, completely ignoring textual input. Reed et al. (2016c) introduced an interesting concept of conditioning DC-GAN (GAN with CNNs as Generator and Discriminator) on textual embeddings. A separate model is being trained and used for encoding the text. Then, result embeddings are concatenated with the noise vector and fed into the Generator and the Discriminator takes embeddings as an input as well. The resulting model is referred to as GAN-INT-CLS. Both abbreviations (INT

and CLS) stand for specific training choices, which are going to be explained later in the chapter. The overview of the proposed architecture can be seen in Figure 3.15.

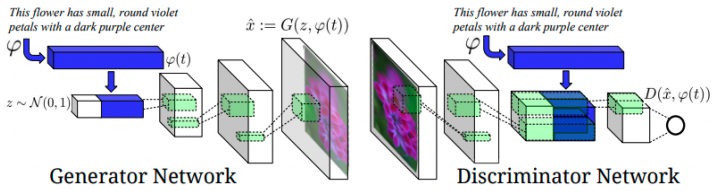


FIGURE 3.15: The proposed architecture of the convolutional GAN that is conditioned on text. Text encoding $\varphi(t)$ is fed into both the Generator and the Discriminator. Before further convolutional processing, it is first projected to lower dimensionality in fully-connected layers and concatenated with image feature maps. Figure from Reed et al. (2016c).

Text embeddings

Since regular text embeddings are commonly trained in separation from visual modality simply by looking at the textual context, they are not well suited for capturing visual properties. This motivated Reed et al. (2016b) to come up with structured joint embeddings of images and text descriptions. GAN-INT-CLS implements it in a way described in Figure 3.16.

The text classifier induced by the learned correspondence function f_t is trained by optimizing the following structured loss:

$$\frac{1}{N}\sum_{n=1}^{N}\Delta(y_n, f_v(v_n)) + \Delta(y_n, f_t(t_n)) \quad (2)$$

where $\{(v_n, t_n, y_n) : n = 1, ..., N\}$ is the training data set, Δ is the 0-1 loss, v_n are the images, t_n are the corresponding text descriptions, and y_n are the class labels. Classifiers f_v and f_t are parametrized as follows:

$$f_v(v) = \arg\max_{y \in \mathcal{Y}} \mathbb{E}_{t \sim \mathcal{T}(y)}[\phi(v)^T \varphi(t))] \quad (3)$$

$$f_t(t) = \arg\max_{y \in \mathcal{Y}} \mathbb{E}_{v \sim \mathcal{V}(y)}[\phi(v)^T \varphi(t))] \quad (4)$$

where ϕ is the image encoder (e.g. a deep convolutional neural network), φ is the text encoder (e.g. a character-level CNN or LSTM), $\mathcal{T}(y)$ is the set of text descriptions of class y and likewise $\mathcal{V}(y)$ for images. The intuition here is that a text encoding should have a higher compatibility score with images of the correspondong class compared to any other class and vice-versa.

FIGURE 3.16: Figure from Reed et al. (2016c).

GoogLeNet is being used as an image encoder ϕ. For text encoding $\varphi(t)$, authors use a character-level CNN combined with RNN. Essentially, the objective of the training is to minimize the distance between the encoded image and text representations. The image encoder is then discarded and φ only is used as depicted in Figure 3.15.

GAN-CLS

CLS stands for Conditional Latent Space, which essentially means the GAN is conditioned on the embedded text. However, in order to fully grasp how exactly the model is conditioned on the input, we need to go beyond architectural choices. It is also crucial to present a specific training regime that was introduced for GAN-CLS and the motivation behind it.

One way to train the system is to view text-image pairs as joint observations and train the discriminator to classify the entire pair as real or fake. However, in such a case the discriminator does not have an understanding of whether the image matches the meaning of the text. This is because the discriminator does not distinguish between two types of error that exist, namely when the image is unrealistic or when it is realistic but the text does not match.

A proposed solution to this problem is to present the discriminator with three observations at a time, all of which are included later in the loss function. These three are: {real image with right text}, {real image with wrong text}, {fake image with right text}. The intention is that the discriminator should classify them as {true}, {false}, {false}, respectively.

GAN-INT

The motivation behind this concept comes from the fact that interpolating between text embeddings tends to create observation pairs that are still close to the real data manifold. Therefore, generating additional synthetic text embeddings and using them instead of real captions in the training process might help in the sense that it works as a form of data augmentation and helps regularize the training process. Figure 3.17 might be helpful for developing the intuition behind the interpolation process.

Results

The model achieves the best performance when both of the mentioned methods are in use (GAN-INT-CLS). Models prove to successfully transfer style (pose of the objects) and background from the training data when trained on CUB (birds) and Oxford-102 (flowers) datasets. They also show interesting zero-shot abilities, meaning they can generate observations from unseen test classes (Figure 3.18). When trained on MS-COCO, GAN-CLS proves its potential to generalize over many domains, although the results are not always coherent (Figure 3.19).

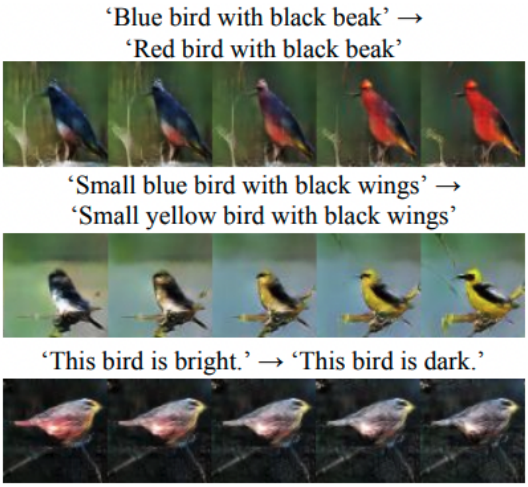


FIGURE 3.17: Interpolating between sentences. Figure from Reed et al. (2016c).

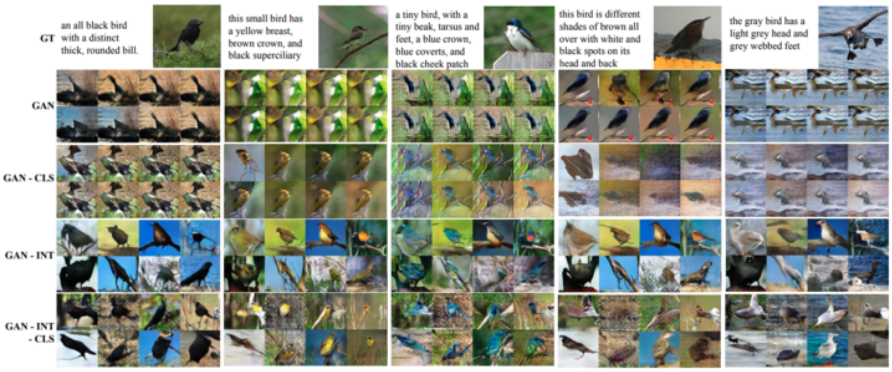


FIGURE 3.18: Zero-shot generated birds using GAN, GAN-CLS, GAN-INT, GAN-INT-CLS. Figure from Reed et al. (2016c).

3.2.2.3 Further GAN-like development

Generative Adversarial Networks were a leading approach for text-to-image models for most of the field's short history. In the following years after the introduction of GAN-INT-CLS, new concepts were emerging, trying to push the results further. Many of them had a GAN architecture as their core part. In this section, a few such ideas are presented. The intention is to quickly skim through the most important ones. A curious reader should follow the corresponding papers.

StackGAN

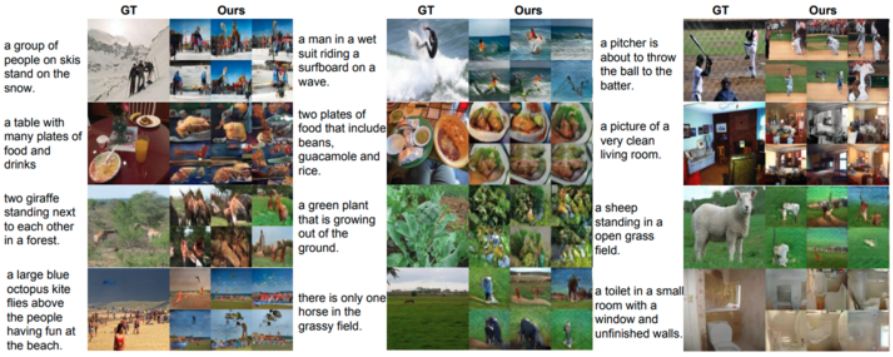


FIGURE 3.19: Generated images using GAN-CLS on MS-COCO validation set. Figure from Reed et al. (2016c).

Zhang et al. (2016a) introduced what the StackGAN. The main contribution of the paper which also found its place in other researchers' works, was the idea to *stack* more than one generator-discriminator pair inside the architecture. Stage-II (second pair) generator is supposed to improve the results from Stage-I, taking into account only:

- text embedding (same as Stage-I)
- image generated in Stage-I

without a random vector. Deliberate omission of the random vector results in the generator directly working on improving the results from Stage-I. The purpose is also to increase resolution (here from 64x64 to 256x256). Authors obtained great results already with two stages, however, in principle architecture allows for stacking many of them.

FIGURE 3.20: (ref:stackgan)

AttnGAN

It is 2017 and many researchers believe attention is all they need (Vaswani

et al., 2017e). Probably for the first time in text-to-image generation attention mechanism was used by Xu et al. (2017). The authors combined the idea with what StackGAN proposed and used three stages (generators G_0, G_1 and G_2). However, this time first layers of a particular generator are attending to word feature vectors. This mechanism not only helps control how particular areas of the image are being improved by consecutive generators but also allows for visualizing attention maps.

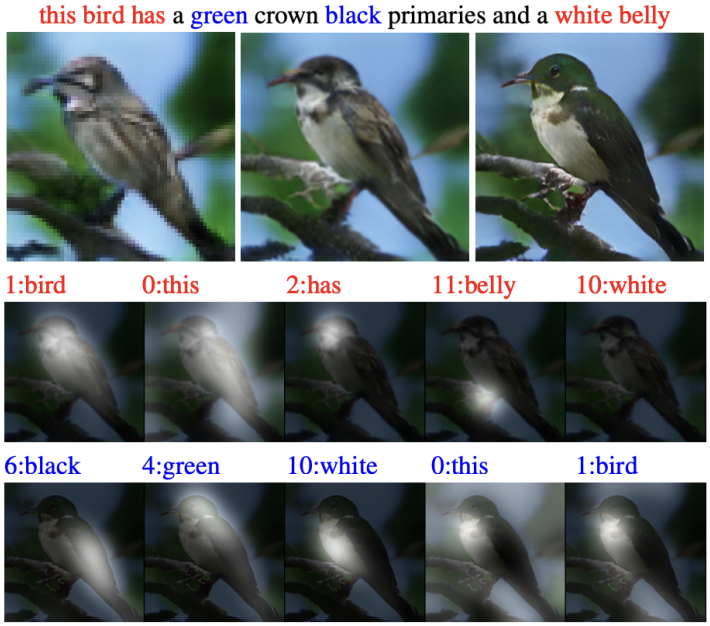


FIGURE 3.21: Images generated by G_0, G_1, G_2. Two bottom rows show 5 most attended words by G_1 and G_2 respectively. Figure from Xu et al. (2017).

DM-GAN

Another important milestone was DM-GAN (Dynamic Memory GAN) (Zhu et al., 2019). At that time, models were primarily focusing on generating the initial image and then refining it to a high-resolution one (as e.g. StackGAN does). However, such models heavily depend on the quality of the first image initialization. This problem was the main motivation for the authors to come up with a mechanism to prevent it. DM-GAN proposes a dynamic memory module, which has two main components. First, its memory writing gate helps select the most important information from the text based on the initial image. Second, a response gate merges the information from image features with the memories. Both of these help refine the initial image much more effectively.

DF-GAN

Last but not least, DF-GAN (Deep Fusion GAN) (Tao et al., 2020) improves the results by proposing three concepts. One-Stage Text-to-Image Backbone focuses on providing an architecture that is capable of abandoning the idea of multiple stacked generators and using a single one instead. It achieves that by a smart combination of a couple of factors, i.a. hinge loss and the use of residual blocks. Additionally, Matching-Aware Gradient Penalty helps achieve high semantic consistency between text and image and regularizes the learning process. Finally, One-Way Output helps the process converge more effectively.

3.2.3 Dall-E 1

OpenAI's Dall-E undoubtedly took the text-to-image field to another level. For the first time, a model showed great zero-shot capabilities, comparable to previous domain-specific models. To achieve that, an unprecedented scale of the dataset and training process was needed. 250 million text-image pairs were collected for that purpose, which enabled training of a 12-billion parameter version of the model. Unfortunately, Dall-E is not publicly available and follows the most recent trend of closed-source models. Or, to put it more precisely, it started this trend, and GLIDE, Dall-E 2, Imagen, Parti and others followed. Nevertheless, Dall-E's inner workings are described in Ramesh et al. (2021b) and this section will try to explain its most important parts. However, before that, it is crucial to understand one of the fundamental concepts that has been around in the field of generative models for already quite some time - namely Variational Autoencoders.

Variational Autoencoder (VAE)

The regular Autoencoder architecture aims at finding an identity function that is capable of finding a meaningful representation of the data in lower-dimensional space and then reconstructing it. It is considered an unsupervised learning method for dimensionality reduction, however, trained in a supervised regime with the data itself being the label. The component performing the reduction is called an encoder, while the part responsible for the reconstruction is called a decoder. The idea behind Variational Autoencoder (Kingma and Welling, 2013) is similar, however, instead of learning the mapping to a static low-dimensional vector, the model learns its distribution. This design equips the decoder part with desired generative capabilities, as sampling from the latent low-dimensional space will result in varying data being generated. The architecture is depicted in Figure 3.22.

$q_\phi(z|x)$ denotes the encoder under the assumption that z comes from multivariate Gaussian. μ and σ are being learned. Reconstruction process is modelled by conditional probability $p_\theta(x|z)$, given samples latent vector z.

VQ-VAE / dVAE

The VQ-VAE (Vector Quantized VAE) (van den Oord et al., 2017) differs from

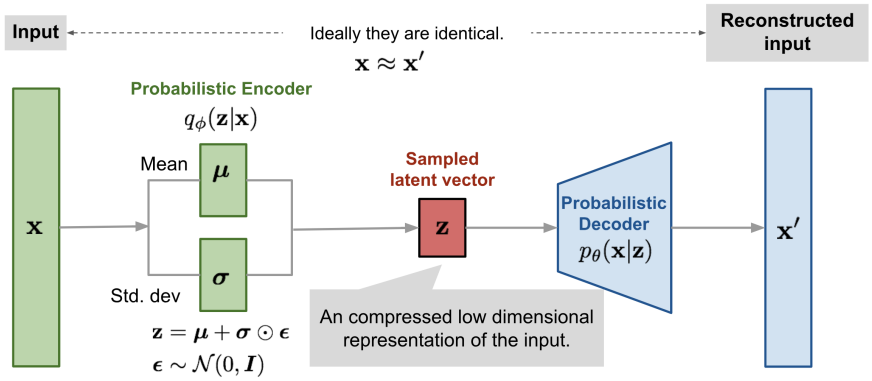


FIGURE 3.22: Variational (probabilistic) Autoencoder architecture. Figure from Weng (2018).

the regular VAE in the way it approaches encoding the latent space. Instead of mapping data into a continuous distribution, the Vector Quantized version does it in a discrete way. This is motivated by the fact that for many data modalities it is more natural to represent them in a discrete way (e.g. speech, human language, reasoning about objects in images, etc.). VQ-VAE achieves that by using a separate codebook of vectors. The architecture is depicted in Figure 3.23.

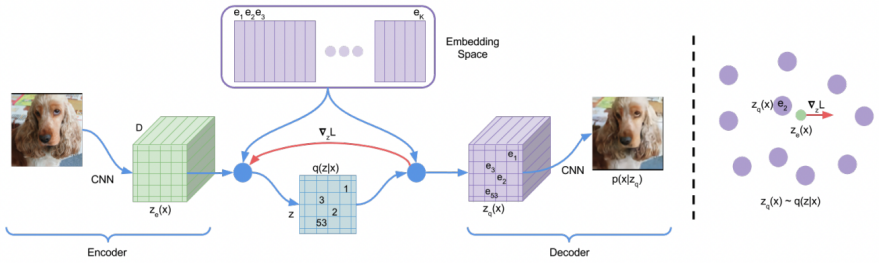


FIGURE 3.23: VQ-VAE architecture. Figure from van den Oord et al. (2017).

The idea is to map the output of the encoder to one of the vectors from the K-dimensional codebook. This process is called quantization and essentially means finding the vector that is the nearest neighbour to the encoder's output (in a sense of Euclidean distance). Since this moment, this newly found vector from the codebook is going to be used instead. The codebook itself is also subject to the learning process. One could argue that passing gradients during the training through such a discrete system might be problematic. VQ-VAE overcomes this problem by simply copying gradients from the decoder's input

to the encoder's output. A great explanation of the training process and further mathematical details can be found in Weng (2018) and Snell (2021).

Dall-E, however, is using what is called dVAE. Essentially, it is a VQ-VAE with a couple of details changed. In short, the main difference is that instead of learning a deterministic mapping from the encoder's output to the codebook, it produces probabilities of a latent representation over all codebook vectors.

Dall-E system

Dall-E is composed of two stages. The above introduction of VQ-VAE was necessary to understand the first one. Essentially, it is training dVAE to compress 256x256 images into a 32x32 grid of tokens. This model will play a crucial role in the second stage.

The second stage is about learning the prior distribution of text-image pairs. First, the text is byte-pair (Sennrich et al., 2015a) encoded into a maximum of 256 tokens, where the vocabulary is of size 16384. Next, the image representation encoded by previously trained dVAE is unrolled (from 32x32 grid to 1024 tokens) and concatenated to the text tokens. This sequence (of 256+1024 tokens) is used as an input for a huge transformer-like architecture. Its goal is to autoregressively model the next token prediction.

During inference time, the text caption is again encoded into 256 tokens at most. The generation process starts with predicting all of the next 1024 image-related tokens. They are later decoded with the dVAE decoder that was trained in the first step. Its output represents the final image.

Results

Results achieved with the original Dall-E attracted so much attention mainly due to its diversity and zero-shot capabilities. Dall-E was capable of producing better results compared to previous state-of-the-art models which were trained on data coming from the same domain as data used for evaluation. One comparison can be seen in Figure 3.24.

Outputs of some of the prior approaches described in this chapter compared with Dall-E can be seen in Figure 3.25.

Limitations

Although Dall-E made a huge step forward in text-to-image modelling, it still showed multiple flaws. First, photorealism of the outputs is still relatively low. In other words, when prompted for images containing realistic situations, it is rarely capable of *deceiving* human evaluators. Second, the model has evident problems with understanding relatively complex abstractions, such as text inside an image, or relative object positions in the scene.

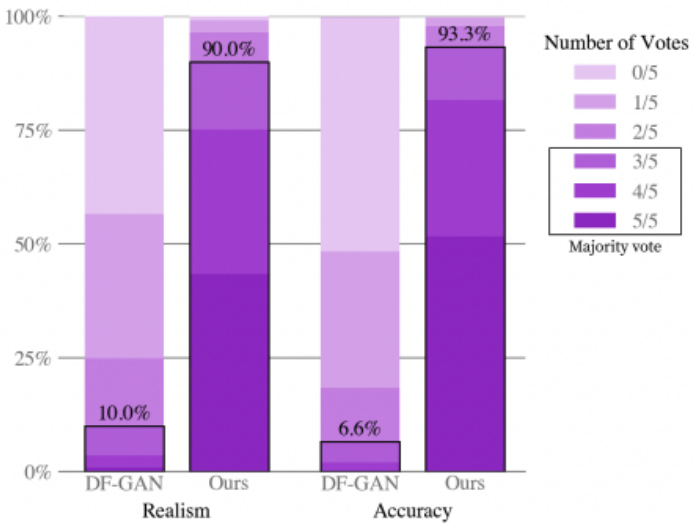


FIGURE 3.24: Human evaluation of Dall-E vs DF-GAN on text captions from the MS-COCO dataset. When asked for realism and caption similarity, evaluators preferred Dall-E's results over 90\% of the time. Figure from Ramesh et al. (2021b).

FIGURE 3.25: Comparison of the results from Dall-E vs prior works on MS-COCO. Dall-E's outputs are chosen as the best out of 512 images, ranked by a contrastive model. Figure from Ramesh et al. (2021b).

3.2.4 GLIDE

Introduced by Nichol et al. (2021b), GLIDE started an era of huge-scale diffusion models. The concept of diffusion has already been used in the area of Deep Learning for some time before. However, the authors of GLIDE took a step further and combined it together with text-based guidance which is supposed to steer the learning process in the direction of the text's meaning. This powerful method was proven to achieve outstanding results which remain competitive with current state-of-the-art models at the time of writing.

Diffusion models

Before understanding the inner workings of GLIDE, it is important to introduce the core concept that is driving it, namely diffusion. The idea of diffusion originates from physics. In short, it corresponds to the process of diffusing particles, for example of one fluid in another. Normally it has a unidirectional character, in other words, it cannot be reversed. However, as Sohl-Dickstein et al. (2015) managed to show, and Ho et al. (2020a) later improved, if the data diffusion process is modelled as a Markov chain with Gaussian noise being added in consecutive steps, it is possible to learn how to reverse it. This reversed process is exactly how images are generated by the model from pure random noise.

Let us construct a Markov chain, where the initial data point is denoted by x_0. In t steps, Gaussian noise is added to the data. The distribution of the data at t-step can be characterized in the following way:

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

where $(1-\alpha_t)$ parametrizes the magnitude of the noise being added at each step. Now, if x_{t-1} was to be reconstructed from x_t, a model needs to learn to predict estimates of gradients from the previous steps. The probability distribution of previous steps can be estimated as follows:

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

where the mean function μ_θ was proposed by Ho et al. (2020a). For a more detailed explanation of how this is later parametrized and trained, one could follow Weng (2021).

GLIDE system

GLIDE can essentially be broken down into two parts. The first of them is the pretrained Transformer model, which in principle is responsible for creating the text embeddings. The last token embedding is used as a class embedding (text representation) in later stages. Additionally, all tokens from the last embedding layer are being used (*attended to*) by all attention layers in the diffusion model

itself. This makes the model aware of the text meaning while reconstructing the previous step in the Markov chain.

The second component of the GLIDE is the diffusion model itself. A U-Net-like architecture with multiple attention blocks is used here. This part's sole goal is to model $p_\theta(x_{t-1}|x_t, y)$, where y corresponds to last token embedding mentioned above. Or, to put it differently, to predict $\epsilon_\theta(x_t|y)$ since the problem can be reframed as calculating the amount of noise being added at each step.

Additionally, to make the model even more aware of the text's meaning, guidance is being used at inference time. In short, the idea is to control the direction of the diffusion process. The authors test two different approaches. First, they try guidance with the use of a separate classifier, OpenAI's CLIP in this case. However, better results were in general achieved by the classifier-free guidance process. The idea is to produce two different images at each step. One is conditioned on text, while the other one is not. Distance between them is calculated and then, after significant scaling, added to the image obtained without conditioning. This way, the model speeds up the progression of the image towards the meaning of the text. This process can be written as:

$$\hat{\epsilon}_\theta(x_t|y) = \epsilon_\theta(x_t|\emptyset) + s * (\epsilon_\theta(x_t|y) - \epsilon_\theta(x_t|\emptyset))$$

where s denotes the parameter for scaling the difference between the mentioned images.

Results

GLIDE achieves significantly more photorealistic results compared to its predecessors. FID scores reported on the MS-COCO 256x256 dataset can be seen in Figure 3.26. It is worth noting that GLIDE was not trained on this dataset, hence its zero-shot capabilities are even more impressing.

Model	FID	Zero-shot FID
AttnGAN (Xu et al., 2017)	35.49	
DM-GAN (Zhu et al., 2019)	32.64	
DF-GAN (Tao et al., 2020)	21.42	
DM-GAN + CL (Ye et al., 2021)	20.79	
XMC-GAN (Zhang et al., 2021)	9.33	
LAFITE (Zhou et al., 2021)	**8.12**	
DALL-E (Ramesh et al., 2021)		~ 28
LAFITE (Zhou et al., 2021)		26.94
GLIDE		**12.24**
GLIDE (Validation filtered)		**12.89**

FIGURE 3.26: Comparison of FID on MS-COCO 256×256. Figure from Nichol et al. (2021b).

Results are also preferred by human evaluators in terms of photorealism and

the similarity of the image to its caption. A comparison to DALL-E 1 results can be seen in Figure 3.27

	DALL-E Temp.	Photo-realism	Caption Similarity
No reranking	1.0	91%	83%
	0.85	84%	80%
DALL-E reranked	1.0	89%	71%
	0.85	87%	69%
DALL-E reranked + GLIDE blurred	1.0	72%	63%
	0.85	66%	61%

FIGURE 3.27: Win probabilities of GLIDE vs DALL-E. Figure from Nichol et al. (2021b).

Finally, some of the cherry-picked images together with their corresponding captions can be seen in Figure 3.28.

FIGURE 3.28: Samples from GLIDE with classifier-free-guidance and s=3. Figure from Nichol et al. (2021b).

Limitations

GLIDE suffers from two problems. First, it fails when being presented with a complex or unusual text prompt. A few examples can be seen in Figure 3.29. Also, the model is relatively slow at inference time (much slower than GANs). This is caused by the sequential character of the architecture, where consecutive steps in Markov chain reconstruction cannot be simply parallelized.

3.2.5 Dall-E 2 / unCLIP

The contribution that probably attracted the most attention in the field is known under the name Dall-E 2 (Ramesh et al., 2022a). For the first time, the wider public had picked interest in its potential applications. This might be due to a great PR that could be seen from the authors, namely OpenAI. Dall-E 2, also known as just Dall-E, or unCLIP, has been advertised as a successor of Dall-E 1, on which results it significantly improved. In reality,

FIGURE 3.29: Failures happen mostly for unusual prompts. Figure from Nichol et al. (2021b).

the architecture and the results it achieved are much more similar to that of GLIDE. Additionally, social media has been flooded with images generated by the model. This was possible thanks to OpenAI giving access to it to everybody who was interested and patient enough to get through a waiting list. However, the model itself again remains unpublished. Another factor that might have contributed to Dall-E's success were its inpainting and outpainting capabilities. Although, it is worth mentioning they were already also possible with GLIDE.

In essence, UnCLIP is a very smart combination of pior work from OpenAI that was re-engineered and applied in a novel way. Nevertheless, the model represents a significant leap forward, which is why it cannot be omitted in this chapter.

Dall-E 2 system

UnCLIP consists of two components: prior and decoder. Let x be the image and y its caption. z_i and z_t are CLIP image and text embedding of this (x, y) pair. Then, *prior* $P(z_i|y)$ is responsible for producing CLIP image embeddings conditioned on the text caption. A decoder $P(x|z_i, y)$ outputs an image conditioned on the CLIP image embedding and, again, the text caption itself.

For the *prior* authors try two different approaches, namely autoregressive and diffusion models. The latter ended up yielding slightly better results. The diffusion prior isa Transformer taking as an input a special sequence of an encoded text prompt, CLIP text embedding, embedding for the diffusion step, and a noised CLIP image embedding.

The decoder consists of diffusion models again. Firstly, a GLIDE-like model takes a CLIP image embedding as its x_t instead of the pure noise that was used in its original version. Similarly to the original GLIDE, classifier-free guidance is applied, however with slight differences. Lastly, two diffusion upsampler models are trained to bring images first from 64x64 to 256x256, and then from 256x256 to 1024x1024 resolution. The authors found no benefit in conditioning

these models on text captions. Finally, unCLIP can be summarized as a mixture of GLIDE and CLIP with a lot of engineering behind it.

Results

When compared to GLIDE, unCLIP shows it is capable of representing a wider diversity of the data, while achieving a similar level of photorealism and caption similarity. Comparison to previous works on the MS-COCO dataset shows that unCLIP achieves unprecedented FID (Figure 3.30). A few output examples calculated on MS-COCO captions can be found in Figure 3.31.

Model	FID	Zero-shot FID	Zero-shot FID (filt)
AttnGAN (Xu et al., 2017)	35.49		
DM-GAN (Zhu et al., 2019)	32.64		
DF-GAN (Tao et al., 2020)	21.42		
DM-GAN + CL (Ye et al., 2021)	20.79		
XMC-GAN (Zhang et al., 2021)	9.33		
LAFITE (Zhou et al., 2021)	8.12		
Make-A-Scene (Gafni et al., 2022)	**7.55**		
DALL-E (Ramesh et al., 2021)		~ 28	
LAFITE (Zhou et al., 2021)		26.94	
GLIDE (Nichol et al., 2021)		12.24	12.89
Make-A-Scene (Gafni et al., 2022)			11.84
unCLIP (AR prior)		10.63	11.08
unCLIP (Diffusion prior)		**10.39**	**10.87**

FIGURE 3.30: Comparison of FID on MS-COCO. The best results for unCLIP were reported with the guidance scale of 1.25. Figure from Ramesh et al. (2022a).

Limitations

UnCLIP suffers from very similar problems as its predecessor GLIDE. First, compositionality in the images tends to sometimes be confused by the model. Failure cases can be seen in Figure 3.32. Second, UnCLIP struggles with generating coherent text inside an image (Figure 3.33). The authors hypothesize that using CLIP embeddings, although improving diversity, might be responsible for making these problems more evident than in GLIDE. Lastly, UnCLIP often fails with delivering details in highly complex scenes (Figure 3.34). Again, according to the authors, this might be a result of the fact that the decoder is producing only 64x64 images which are later upsampled.

3.2.6 Imagen & Parti

Only a few months after unCLIP was released by OpenAI, for the first time Google came into play with its new autoregressive model called Imagen (Saharia et al., 2022b). Another one followed just two months later - Parti (Yu et al., 2022b). Both of these models pushed the boundaries even further, although they take entirely different approaches. None of them is introducing a completely new way of looking at the problem of text-to-image generation. Their advancements

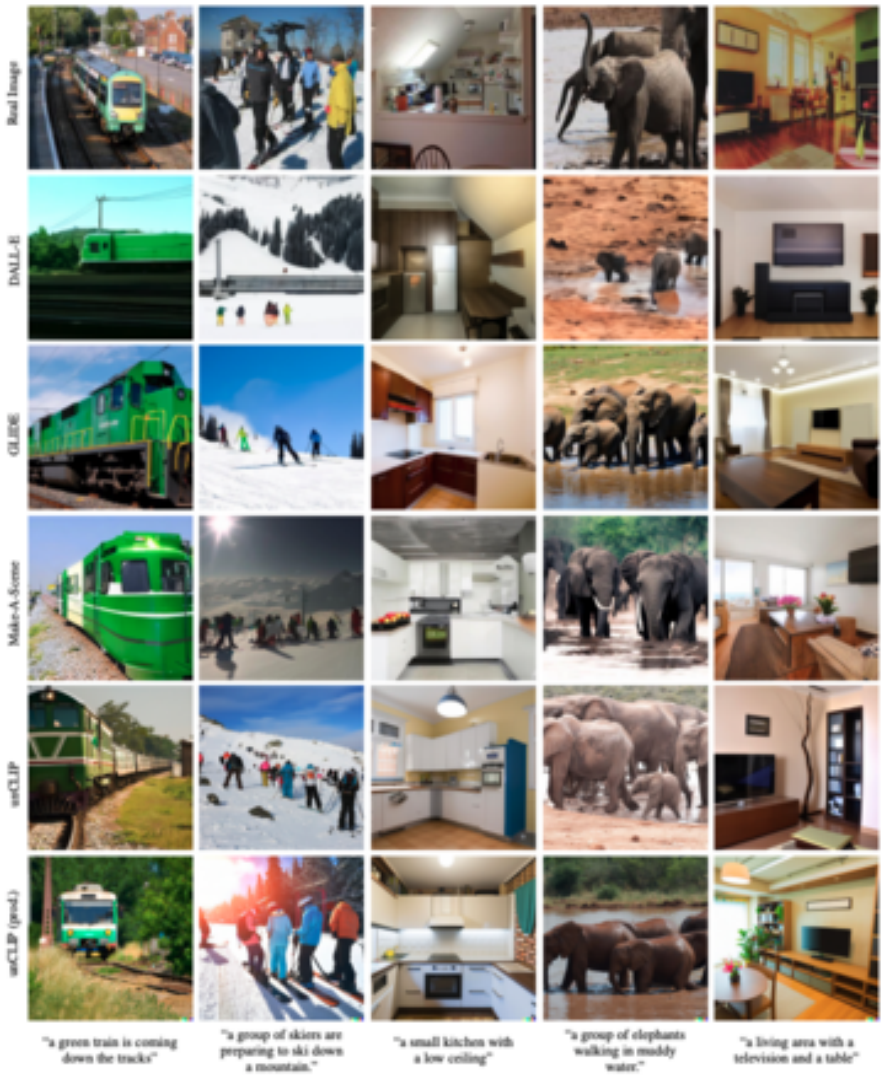
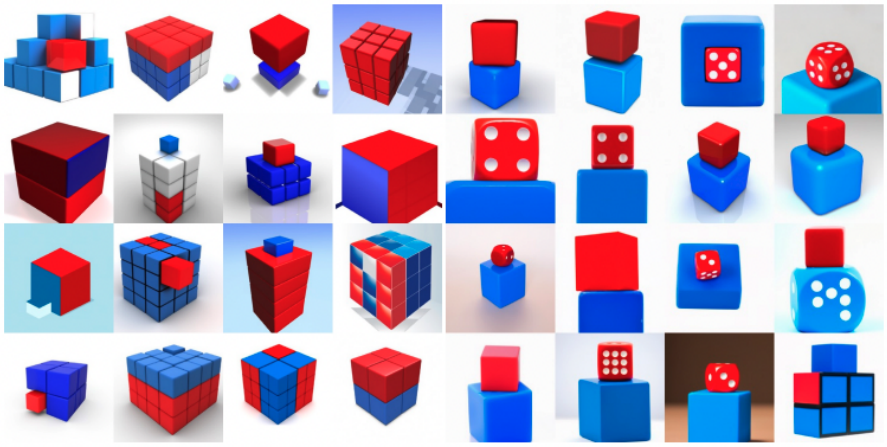


FIGURE 3.31: Image samples on MS-COCO text prompts. Figure from Ramesh et al. (2022a).



(a) unCLIP (b) GLIDE

FIGURE 3.32: 'a red cube on top of a blue cube' Figure from Ramesh et al. (2022a).

FIGURE 3.33: 'A sign that says deep learning.' Figure from Ramesh et al. (2022a).

come from engineering and further scaling existing solutions. However, it must be stressed that currently (September 2022) they are delivering the most outstanding results.

Imagen is a diffusion model. Its main contribution is that instead of using a text encoder trained on image captions, it actually uses a huge pretrained NLP model called T5-XXL (Raffel et al., 2019b) that is taken off the shelf and frozen. Authors argue that this helps the model understand language much more deeply, as it has seen more diverse and complex texts than just image captions.

On the other hand, Parti takes an autoregressive approach. Similarly to the first version of Dall-E, it consists of two stages, namely the image tokenizer and sequence-to-sequence autoregressive part which is responsible for generating image tokens from a set of text tokens. In this case, ViT-VQGAN (Yu

FIGURE 3.34: 'A high quality photo of Times Square.' Figure from Ramesh et al. (2022a).

et al., 2021) is used as a tokenizer and the autoregressive component is again Transformer-like.

Results

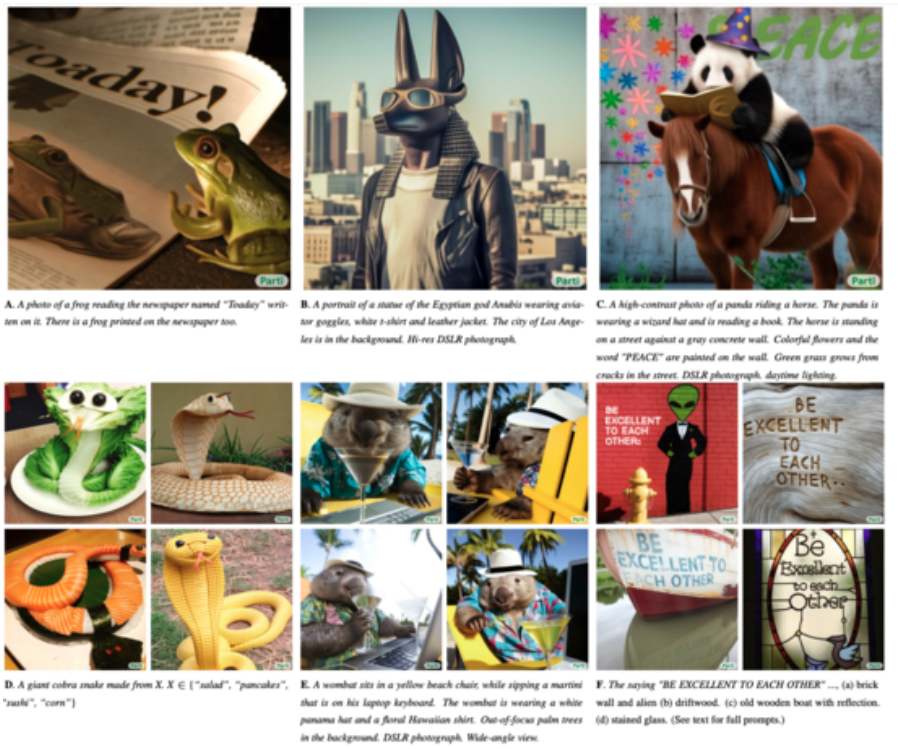
Both of the models improved the FID significantly compared to the previous works. Figure 3.35 shows the comparison.

Approach	Model Type	MS-COCO FID (↓)		LN-COCO FID (↓)	
		Zero-shot	Finetuned	Zero-shot	Finetuned
Random Train Images [10]	-	2.47		-	
Retrieval Baseline	-	17.97	6.82	33.59	16.48
TReCS [46]	GAN	-	-	-	48.70
XMC-GAN [47]	GAN	-	9.33	-	14.12
DALL-E [2]	Autoregressive	∼28	-	-	-
CogView [3]	Autoregressive	27.1	-	-	-
CogView2 [61]	Autoregressive	24.0	17.7	-	-
GLIDE [11]	Diffusion	12.24	-	-	-
Make-A-Scene [10]	Autoregressive	11.84	7.55	-	-
DALL-E 2 [12]	Diffusion	10.39	-	-	-
Imagen [13]	Diffusion	**7.27**	-	-	-
Parti	Autoregressive	**7.23**	**3.22**	**15.97**	**8.39**

FIGURE 3.35: Comparison of FID on MS-COCO. Figure from Yu et al. (2022b).

Samples from Parti can be seen in Figure 3.36. They are included here on purpose - this is the current state-of-the-art as of the moment of writing!

Limitations



A. *A photo of a frog reading the newspaper named "Toaday" written on it. There is a frog printed on the newspaper too.*

B. *A portrait of a statue of the Egyptian god Anubis wearing aviator goggles, white t-shirt and leather jacket. The city of Los Angeles is in the background. Hi-res DSLR photograph.*

C. *A high-contrast photo of a panda riding a horse. The panda is wearing a wizard hat and is reading a book. The horse is standing on a street against a gray concrete wall. Colorful flowers and the word "PEACE" are painted on the wall. Green grass grows from cracks in the street. DSLR photograph, daytime lighting.*

D. *A giant cobra snake made from X. X ∈ {"salad", "pancakes", "sushi", "corn"}*

E. *A wombat sits in a yellow beach chair, while sipping a martini that is on his laptop keyboard. The wombat is wearing a white panama hat and a floral Hawaiian shirt. Out-of-focus palm trees in the background. DSLR photograph. Wide-angle view.*

F. *The saying "BE EXCELLENT TO EACH OTHER" ...,* (a) brick wall and alien (b) driftwood. (c) old wooden boat with reflection. (d) stained glass. (See text for full prompts.)

FIGURE 3.36: Selected samples from Parti. Figure from Yu et al. (2022b).

Yu et al. (2022b) mention an extensive list of problems, with which Parti still struggles. At this point, all of them can be treated as a set that is common to almost all available models. Among others, they touch:

- feature blending (where features of two different objects are missed)
- omission or duplicating details
- displaced positioning of objects
- counting
- negation in text prompts

and many many more. These flaws pose a challenge for future research and undoubtedly they are the ones that need to be addressed first to enable another leap forward in the field of text-to-image generation.

3.2.7 Discussion

Lastly, it is important to mention a couple of different topics, or trends, which are intrinsically linked with text-to-image generation. Together with previous

sections, they should give the reader a holistic view of where research currently stands (again, as of September 2022).

Open- vs closed-source

The first trend that has emerged only recently is AI labs to not open-source their state-of-the-art models and training data. This is in clear opposition to how the entire AI community was behaving from the very beginning of the recent Deep Learning era Apparently, possible commercial opportunities that come along with owning the software are too big to be ignored. The trend is very disruptive - it is clear that the community is currently witnessing the maturation of AI business models. Needless to say, it is followed by all the greatest AI labs, just to name a few: OpenAI, DeepMind, Google Brain, Meta AI, and many others. As long as commercial achievements will have an edge over academic community research, it is highly doubtful that the trend will be reversed. However, it needs to be stressed that all of them are still issuing more or less detailed technical specifications of their work in the form of scientific papers, which is definitely a positive factor. We, as a community, can only hope it will not change in the future.

Open-Source Community

As the trend of closed-sourceness is clearly visible across many Deep Learning areas, the text-to-image research is actually well represented by an open-source community. The most important milestones of the recent years indeed come from OpenAI, however, new approaches can be seen across a wide community of researchers. Many of these models are public, meaning that any user with minimal coding experience can play with them. Although we decided not to go into details of particular works, it is important to name a few that became the most popular:

- VQGAN-CLIP (Crowson et al., 2022)
- Midjourney (Midjourney, 2022)
- Latent Diffusion (Rombach et al., 2021)
- Stable Diffusion (Rombach et al., 2022)

Potential applications

Image generation that can be done in a controllable manner has undoubtedly huge potential for commercialization. Although the field is currently still very immature, hypotheses about which industries might be disrupted are emerging. Essentially, every branch that has to do with generating visual art, be it static images or videos, should observe the trend closely. Graphic design, movie making, stock photos - just to name a few that might be interested. Currently, experimental use cases in the area of texture synthesis, product design, or building virtual reality worlds can already be observed. AI, even if still incapable of generating the final product, can help automate a significant part of the production chain, which essentially means time and money savings.

The inpainting and outpainting capabilities of recent models play a significant role in this trend. Although it is still very hard to judge which direction it takes in the future, it will definitely be a very interesting and disruptive change. Who wouldn't like to see movies being soon generated directly from a book's text, pixel value by pixel value?

Ethics / Conclusion

Automated image generation poses an array of serious questions of ethical character. Fortunately, many of them are already very well recognized by the community. For example, OpenAI elaborates extensively on the risks and limitations of their Dall-E 2 in this blog post by Mishkin et al. (2022). A few of the most important topics are presented here.

The first and very significant risk is the potential misuse of the models. Fake image generation can easily be used for harassment and disinformation. Especially combined with inpainting, which is capable of erasing or adding objects to real scenes, it poses a non-trivial challenge for researchers on how to responsibly share their work.

Another important area touches on biases and stereotypes which are intrinsically built into the technology. Obviously, a model combines concepts from the data it has seen. However, if this area is to be commercialized, it needs to ensure broader diversity. An interesting example of Dall-E 2 samples can be seen in Figure 3.37.

In order to fully enable AI generation, the problem of copyrights needs to be solved in the first place. It is definitely not clear who is the author of generated images. Is it the person who came up with a text prompt and ran the model? Is it a model engineer? The author of the model's architecture? The owner of the data it has been trained on? Or maybe the model itself? Another question is what really is a creative contribution and eventually should result in copyright being granted. These and many others definitely require extensive debate and hopefully, legal solutions following it.

3.3 Images supporting Language Models

Author: Giacomo Loss

Supervisor: Matthias Aßenmacher

3.3.1 Words In (Non-Symbolic) Contexts

Imagine you were alone in a foreign country, you could not speak the language and the only resource you had were a dictionary in the foreign language. You see

FIGURE 3.37: Biased samples from Dall-E 2. Figure from Mishkin et al. (2022).

a word written on a sign but you cannot understand its meaning. What could you do? One idea would be do open the dictionary and look the word up. The problem is that the word is defined by using other words in the foreign language. As a second step you would thus look these new words up and continue like that in further steps to the "infinity and beyond" (cit. Buzz Lightyear). But even after looking every single word in the dictionary up, you would still not be able to understand the meaning of the word written on the sign. If on that sign, next to the unknown word, something else was instead depicted, for example an image of a fork and a knife, you might speculate that the word indicates something which has to do with food, like a restaurant. And this without explicitly knowing the meaning of the word. This example is inspired by the work of Stevan Harnad, which formulated at the beginning of the 90's the so called *Symbol Grounding Problem* (Harnad (1990)). It asserts that it is not possible to understand the meaning (semantics) of a word by just looking

at other words because words are essentially meaningless symbols. It is possible to understand the meaning only if the word is put in a context, a perceptual space, other than that of written language: the word must be *grounded* in non-symbolic representations, like images, for example. Over the past 10 years there has been a whopping development of distributional semantic models (DSMs, henceforth), especially after the Word2vec (Mikolov et al. (2013b)) revolution. This family of models assumes that the meaning of words and sentences can be inferred by the "distribution" of those words and sentences within a text corpus (the *Distributional Hypothesis* formulated by Harris et al. (1954)). But the *Symbol Grounding Problem* mentioned earlier suggests that DSMs do not resemble the way words are learned by humans, which is in multimodal perceptual contexts. For these reasons, models have been developed with the goal to integrate further modalities (like visual ones) in pure language models, assuming that grounding words and sentences in other perceptual contexts should lead to a better understanding of their semantics and, as a result, to better performance in pure language tasks.

The focus of this subchapter are models which empower pure language models with visual modalities in form of images: their goal is to obtain better semantic representations (in form of embedding vectors) of words. First, a quick recap of the main pure language models will be provided. After that, the historical evolution of the integration of images as visual modalities into pure language models will be discussed: from simple concatenation of textual and visual modalities, to the projection of visual elements in a common grounded space and more recently, the use of Transformers (see figure 3.38). Eventually, a comprehensive evaluation of the different models against benchmarks will be carried out.

Again, the focus is on how to employ visual elements to obtain embeddings able to capture the semantics of words. More concrete applications, such as those in the field of machine translation are out of scope and will be only marginally addressed at the end of the subchapter.

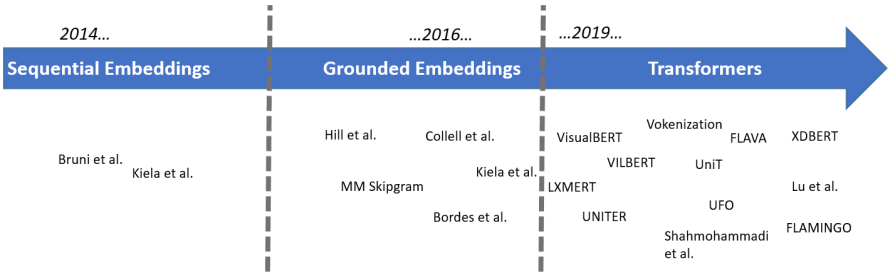


FIGURE 3.38: Historical evolution of models which integrate visual information into pure language models.

3.3.2 Word-Embeddings: Survival-Kit

In other parts of this books, the most important NLP-models and the latest developments in the field are extensively described. In this section, some information will be provided, which might be helpful to understand some of the aspects discussed in this subchapter. As it may have been inferred in the introduction, the starting point is always a pure language model, namely a model which employs only textual inputs in order to generate word embeddings, which are representations of words in form of numerical vectors. The most widely used pure language models in the papers presented in this subchapter are the following three:

- **Skipgram** (Word2vec, Mikolov et al. (2013b)), where given a target word, the probability of the neighboring (surrounding) words in a pre-defined window has to be maximized. Trainig takes place either through a *hierarchical softmax* or through *negative sampling*, which involves maximizing the probability of words which are real neighbors and minimizing that of words which are not real neighbors (the "negative samples")
- **GloVe** (Pennington et al. (2014)), which is based on words co-occurrence across the *entire* corpus, with the goal of minimizing the difference between the dot product of the embedding vectors of two words and the logarithm of the number of co-occurrences
- **BERT** (Devlin et al. (2018c)): two pre-training tasks to obtain word-embeddings:
  - Masked Language Modelling (MLM): given a sentence with [MASK]ed tokens, the goal is to predict these masked tokens
  - Next Sentence Prediction (NSP): given two sentences A and B, the goal is to predict if B follows from A

Two additional remarks to conclude this section. First, Skipgram and GloVe generate embeddings which are *"context-free"*: they do not take into account the context in which words occur. On the contrary, BERT is designed to represent words given the context (sentence) in which they occur: we can thus have different embeddings for the same word, depending on the context. Second, the inputs of these models are *tokens*: with the help of a *tokenizer*, which can be different for different models, the text is split in "chunks", called *tokens* (and they are not necessarily single words).

3.3.3 The Beginning: Sequential Multimodal Embeddings

Supposing we add linguistic and visual feature representations related to a particular word, how could we fuse them? One intuitive idea would be to *concatenate* the textual and visual modalities. Let V_{text} be the textual (vectorial) representation of a word and let V_{img} be its visual (vectorial) representation, a fused representation F of a certain word w might take the following simplified form:

$$F = \gamma(V_{text}) \bigoplus (1-\gamma)V_{img}$$

where γ is a tuning parameter which controls the relative contribution of both modalities to the final fused representation. Bruni et al. (2014) propose a model where the meaning of a target word is represented in the form of a semantic vector and all vectors are collected in a *text-based semantic matrix*; textual embeddings are computed based on (transformed) co-occurrence counts of words in a pre-defined window. The starting point to obtain an image-based representation of certain target word is a dataset of labeled images. For each image associated to the target word (which means that the target word is to be found in the image's caption), low-level features called "local descriptors" - which incorporate geometric information of specific areas of a certain picture - are extracted and then these descriptors are assigned to clusters (*bags*) of "visual words"[1]. Afterwards, for each target word, visual word occurrences are summed up together to obtain the occurrence counts related to the target word. These image-based semantic vectors are then transformed and collected in an *image-based semantic matrix*. The two matrices are then concatenated and projected into a common latent multimodal space with a singular value decomposition. Thanks to this process a *textual* ***mixed*** *matrix* and a *visual* ***mixed*** *matrix* are extracted and then combined together according to different fusion strategies to build the multimodal embeddings. In this first, relatively cumbersome (historically motivated) example, the vector representation of an image is obtained with non-trivial features engineering.

In recent years, the use of neural networks has made an "automatic feature selection" possible. This is what for example Kiela and Bottou (2014) propose, extracting visual features from the first seven layers of a convolutional neural network (proposed by Krizhevsky et al. (2012b)) trained on 1.6 million images from the ImageNet database (Deng et al. (2009)), which produces scores for 1,512 object categories. The linguistic part of the model relies on the Skipgram model by Mikolov et al. (2013b) and consists of 100-dimensional vector representations. The multimodal representation is again obtained by concatenation of both modalities.

Another notable example of concatenation/sequential combination of textual and visual modalities is the work of Silberer and Lapata (2014): textual and visual modalities are represented by separate vectors of textual and visual attributes. During training, these textual and visual inputs vectors are separately fed to denoising (unimodal) autoencoders, the training objective of which is the reconstruction of a certain corrupted input - e.g. through masking noise - from a latent representation. Their outputs are then jointly fed to a bimodal autoencoder to be mapped to a multimodal space, on which

[1] See for example Bosch et al. (2007) for more details on this technique, called "bag-of-visual-words".

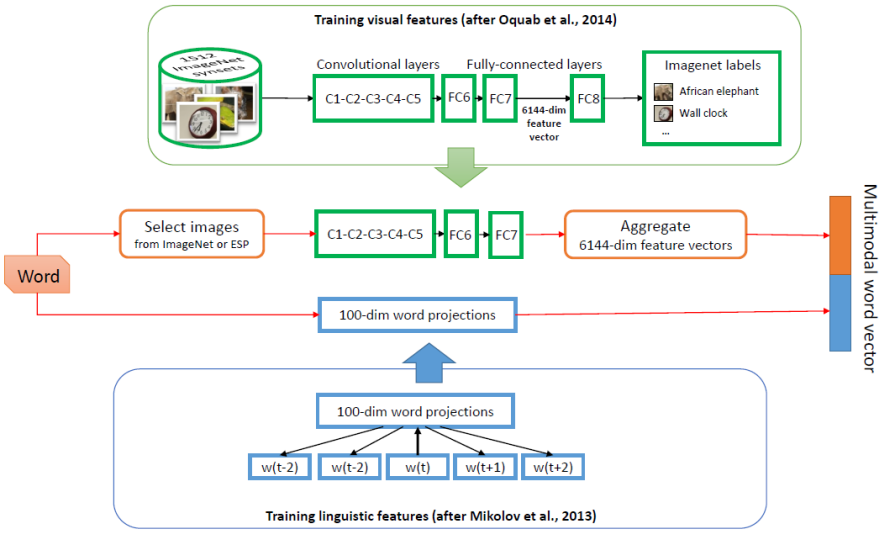


FIGURE 3.39: From Kiela and Bottou (2014). Textual and visual features vectors are concatenated.

a softmax layer (classification layer) is added, which allows the architecture to be fine-tuned for different tasks.

3.3.4 The Grounded Space

The aforementioned models assume implicitly a one-to-one correspondence between text and images: a visual representation is extracted only from words which are associated to a concrete image. This is a limitation, for two partially overlapping reasons. One one hand, how can we depict words for which no image is available in our training set? Is it possible to *imagine* visual representations purely from linguistic ones? On the other hand, could we hypothetically find a visual representation for each word? This might be true for concrete words but when it comes to abstract ones, it is not always possible to find suitable visual representations or, said in other terms, many words are not visually grounded. For this reasons, researches have addressed the question: could we map textual and visual elements to a grounded space and design models able to generalize images and words beyond those in the training set? Well, the answer is yes!

Lazaridou et al. (2015) propose a multimodal Skip-gram architecture where the objective function of a Skip-gram is "augmented" with an additional visual objective:

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{L}_{ling}(w_t)+\mathcal{L}_{vision}(w_t)\right)$$