# Chapter 7
# Foundation Models for Speech, Images, Videos, and Control


**Abstract** Foundation Models are able to model not only tokens of natural language but also token elements of arbitrary sequences. For images, square image patches can be represented as tokens; for videos, we can define tubelets that span an image patch across multiple frames. Subsequently, the proven self-attention algorithms can be applied to these tokens. Most importantly, several modalities like text and images can be processed in the same sequence allowing, for instance, the generation of images from text and text descriptions from video. In addition, the models are scalable to very large networks and huge datasets. The following multimedia types are covered in the subsequent sections. Speech recognition and text-to-speech models describe the translation of spoken language into text and vice versa. Image processing has the task to interpret images, describe them by captions, and generate new images according to textual descriptions. Video interpretation aims at recognizing action in videos and describing them through text. Furthermore, new videos can be created according to a textual description. Dynamical system trajectories characterize sequential decision problems, which can be simulated and controlled. DNA and protein sequences can be analyzed with Foundation Models to predict the structure and properties of the corresponding molecules.

**Keywords** Speech recognition · Text-to-speech · Image captioning · Text-to-image · Video interpretation · Robot control · DNA

Astonishing results of Foundation Models in natural language tasks have led the multimedia processing community to study their application to speech recognition and computer vision problems. Among the most important advantages of Foundation Models is that they can model long dependencies between elements of the input sequence and support parallel processing of the sequence in contrast to recurrent networks. Unlike convolutional networks, Foundation Models require minimal restrictions in the modeling of dependencies and are able to define maps between high-dimensional quantities. In addition, the simple design of Foundation Models allows simultaneous processing of multiple modalities (e.g., images, videos, text and speech) using similar processing blocks. Moreover, the models are scalable



G. Paaß, S. Giesselbach, *Foundation Models for Natural Language Processing*,
Artificial Intelligence: Foundations, Theory, and Algorithms,
https://doi.org/10.1007/978-3-031-23190-2_7

to very large networks and huge datasets. These strengths of Foundation Models have led to comprehensive advances on a number of multimedia tasks.

We will describe multimedia applications in five areas and we will review the currently best approaches, taking into account necessary resources, e.g. computation and memory effort.

- *Speech* recognition and text-to-speech models (Sect. 7.1).
- *Image* description by text and generating images from text (Sect. 7.2).
- *Video* interpretation and video generation (Sect. 7.3).
- *Dynamical system trajectories* describe sequential decision problems, which can be simulated and controlled (Sect. 7.4).
- *DNA and protein sequences* can be analyzed with Foundation Models to predict the structure and properties of the corresponding molecules (Sect. 7.5).

In addition, there are a number of applications, where several media types are processed simultaneously. There is a large list of more specialized media types, where multimodal PLMs have been used: tables [25], text layout [61], depth images [119], scene graphs [60], SQL [18], sign language [199], point cloud [197], symbolic knowledge graph [4], multimodal knowledge graph [201], abstract syntax tree [202], optical flow [50], etc. Processing these media types with Foundation Models is similar to the approaches described in the following sections.

Due to the enormous number of different Foundation Models in the literature, we focus on representative models that have high performance at the time of writing. We outline the inner logic and main features of the methods, taking into account the resources required, e.g., computational and memory requirements. For standard PLMs, a link to descriptions in earlier chapters is provided. Xu et al. [183] compiled a survey on multimodal learning with transformers. Under the heading "Available Implementations" we list links to available code and pre-trained models for that task. Good sources for code are the websites https://paperswithcode.com/, the NLP index https://index.quantumstat.com/, and GitHub https://github.com/github. Processing these media types with PLMs is similar to the approaches described in the following sections.

## 7.1 Speech Recognition and Generation

Spoken language is the most efficient and natural type of communication between humans. Therefore, it is also a preferred type of interaction with computer systems. In the next sections we describe advanced models for automatic speech recognition and text-to-speech systems.

### 7.1.1 Basics of Automatic Speech Recognition

*Automatic speech recognition* (*ASR*) receives a speech input as an audio file and converts it into natural language text. Speech is strongly influenced by gender, social style, dialect, speaking style, and speed. Human speech and accents vary widely, and these differences in speech patterns are one of the major obstacles in developing an automatic speech recognition system. Another impediment to the development of an ASR is finding sufficient training collections to train the ASR model. Currently, training data is available for only a few of the approximately 7000 world languages.

Since the advent of the computer in the 1950s, researchers started to develop speech recognition systems. In 1984, IBM introduced the first speech recognition system that could recognize about 5000 individual English words, and in 1993, a consumer ASR was offered. The predominant techniques were $n$-gram models, hidden Markov models, and neural networks [102]. After 2010, speech recognition based on RNNs was widely used for virtual assistants like Apple's Siri, Amazon Alexa, and Google Assistant. Meanwhile, ASR is in use on most smartphones to enter text by voice even without an Internet connection.

The most important evaluation measure of ASR systems is the *word error rate* $\text{WER} = \frac{S+D+I}{N}$ measuring the deviation from a ground truth text. Here $S$ is the number of word substitutions, $D$ is the number of deletions, and $I$ is the number of insertions in the output as compared to the ground truth with $N$ words.

Conventional ASR systems usually consist of independent parts, such as an acoustic model, a pronunciation model, and a language model. These parts are trained separately and then combined for inference. Usually, a pre-processing module is employed to reduce the signal-to-noise ratio in the audio recording. There are different filters and methods that can be applied to a sound signal to reduce the associated noise. In addition, the speaker may be recorded with several microphones, which can localize the speaker and drastically reduce background noise (beamforming) [24].

Subsequently, a feature extraction module has the task to generate features relevant for speech recognition, remove irrelevant information from the signal and reduce the input size. This often involves variants of Fourier transforms extracting the frequency of waveforms. Most commonly used feature extraction methods are *Mel Frequency Cepstral Coefficients* (MFCCs), discrete wavelet transform (DWT), and linear predictive coding (LPC) [101]. An example is shown in Fig. 7.1.

The final module is a classifier receiving a vector of fixed length characterizing the signal in the given time slot. It estimates the probability of output words or phonemes for the next time slot. Early classifiers could only handle a single speaker. New models were developed to recognize the speech utterances of multiple speakers. An example is an ASR system yielding a 5.1% word error rate (WER) on the switchboard test set [181]. It consists of CNN models like ResNet and LACE and bidirectional LSTMs for modeling acoustics. A survey of prior systems is provided by Malik et al. [101]. A survey of more recent ASR systems is given by Papastratis [117], who discuss RNN, CNN and Transformer models.

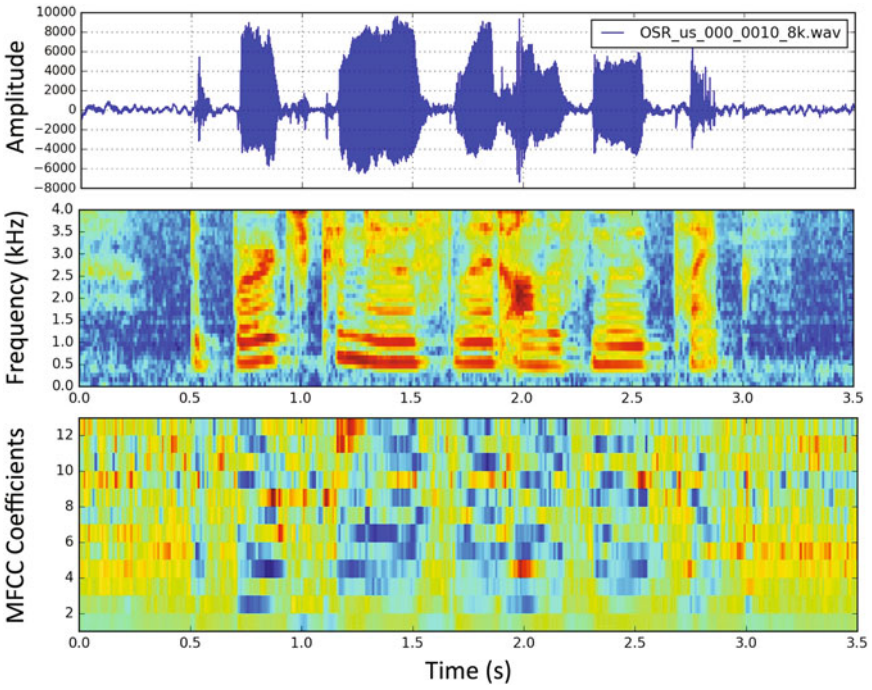


**Fig. 7.1** Audio signal (top) with the frequency extracted by Fourier transform (middle) and the corresponding MFCCs (bottom). Image credits in Table A.3

### *7.1.2 Transformer-Based Speech Recognition*

PLMs based on self-attention are a good choice for sequence modeling because they are able to capture interactions over long distances and require less computational effort. An overview is given in Table 7.1. However, PLMs are less capable of extracting fine-grained local feature patterns. Therefore, combinations of PLMs and CNNs are often used for ASR. The currently best LSTM-based ASR system **ContextNet + NST** [121] achieved an WER of 1.7% on LibriSpeech (clean).

The **Conformer** [59] is a convolution-augmented Transformer. The Conformer integrates a convolutional module (Sect. 1.7) and a self-attention module (Sect. 2.3) as layers inside an encoder block. The convolution module contains a $1 \times 1$ pointwise convolution with an expansion factor of 2 projecting the number of channels with a *Gated Linear Unit* (GLU) activation layer, which allows the selection of features that are important for prediction. This is followed by a *1-D depthwise convolution*, which applies a single convolutional filter for each input channel. Subsequently, there is a batch normalization and then a *Swish* [131] activation layer.

The resulting model with 17 conformer blocks has up to 118M parameters and is trained on the *LibriSpeech* [116] dataset, which contains audiobooks spoken by

**Table 7.1** Main speech recognition techniques

| Model | Mechanism | Performance |
|---|---|---|
| ContextNet + NST | Currently best LSTM-based ASR system | Librispeech WER 1.7% |
| Conformer | CNN + self-attention in transformer block, LSTM as language model | Librispeech WER 1.9% |
| wav2vec 2.0 | Encode speech by CNN, discretize input to transformer, predict masked input. Fine-tune for speech recognition | Librispeech WER 1.5% |
| Combined SSL | Conformer model + unsupervised wav2vec 2.0, SpecAugment to generate noisy training data | Librispeech WER 1.4% |
| SpeechStew | Similar to Combined SSL, trained on 7 datasets, fine-tuned for speech recognition | Librispeech WER 1.7% without Language model |

different speakers. It gets a vector of 80 filterbank features (Fig. 7.1) for each time slot of 10ms. The authors use SpecAugment [120] masking varying parts of the input signal to regularize the model. In addition, they train a 3-layer LSTM language model on the LibriSpeech corpus predicting the next word. The output of the language model is combined with the transformer output to emphasize words which are syntactically and semantically correct. Together with the LM the Conformer achieves a WER of 1.9% on LibriSpeech (clean). Without LM the WER was 2.1%.

The **S4** [58] model is able to process long input sequences of up to 16k elements (Sect. 3.2.2). It was applied to speech classification and was able to improve SOTA to 98.3% while processing raw speech signals. This is an enormous error reduction compared to the prior SOTA accuracy of 95.3%. It can be expected that this model will also lead to a considerable reduction of errors in other speech recognition tasks.

### *7.1.3 Self-supervised Learning for Speech Recognition*

Self-supervised learning of speech has the potential to enhance speech recognition results with additional unlabeled data. It can be shown that self-training on a large set of unlabeled data leads to a strong improvement of models which achieve superior performance with relatively little fine-tuning data [184].

**wav2vec 2.0** [10] performs unsupervised learning on speech data without transcripts. Similar to the BERT model for text, it learns to predict masked sound "tokens". wav2vec encodes raw speech audio by a multi-layer CNN yielding a latent representation of speech for every time slot. The continuous latent representation is discretized to tokens $\boldsymbol{q}_t$ with a quantization module. This discretization is a discontinuous operation and hinders gradient backpropagation.

One solution is to use an interpolation between the discrete result of sampling and the probability distribution. This can be achieved with the *Gumbel-Softmax distribution* [75]. To sample a discrete distribution with probabilities $p_1, \ldots, p_k$

we can draw a random uniform variable $U \sim \text{uniform}(0, 1)$ and compute $Z = \text{onehot}(\max_i p_1 + \cdots p_{i-1} \leq U)$, where $i = 1, \ldots, k$ is the discrete index, and $\text{onehot}(j)$ generates a vector of zeros with a one at position $j$. This sampling is not differentiable because of the max function. An alternative formula is

$$Z = \text{onehot}(\text{argmax}_i(G_i + \log(p_i))), \tag{7.1}$$

where $G_i \sim \text{Gumbel}(0, 1)$ are i.i.d. samples drawn from the standard Gumbel distribution. This refactors the sampling of $Z$ into a deterministic function of the parameters and some independent noise with a fixed distribution. Now a softmax function can be used as a differential approximation of argmax:

$$y_i = \frac{\exp((G_i + \log p_i)/\tau)}{\sum_j \exp((G_j + \log p_j)/\tau)}. \tag{7.2}$$

$\tau$ is the temperature parameter that controls how closely the new samples approximate the discrete vectors. This approximation is used during training and the discretized onehot vectors are computed during evaluation. wav2vec computes discrete vectors $\boldsymbol{q}_t$ by this approach.

The $\boldsymbol{q}_t$ representations of 10 randomly sampled consecutive time steps are masked and have to be reconstructed by a Transformer similar to BERT. The self-attention captures dependencies over the entire sequence of latent representations. This model was pre-trained on more than 1000h of labeled and unlabeled speech data. The pre-trained model is fine-tuned for speech recognition by adding a randomly initialized linear projection on top of the context network into $C$ classes, which were the characters as well as a word boundary marker. To accommodate characters spanning several time slots the *connectionist temporal classification* (CTC) loss [57] was employed. The fine-tuning used 5h of audio data annotated with phonemes. On LibriSpeech the authors achieve a WER of 2.1%. A similar model with 300M parameters using 53k hours of unlabeled data for wave2vec and 10m of labeled data for fine-tuning achieves a WER of 3.0% on LibriSpeech [184]. Training on all data decreases WER to 1.5%.

**Combined SSL** [196] combines wave2vec unsupervised pre-training with the Conformer. The ASR network is a sequence 'translator' consisting of a Conformer encoder with up to 1B parameters and a multilayer LSTM decoder. In addition, the authors use Noisy Student Training (NST), where a teacher model is employed to generate transcripts for the unlabeled data via inference on audio. The teacher-labeled data, after filtering and balancing, are then used to train the next generation ASR model. On LibriSpeech the model achieves SOTA with 1.4% WER.

**w2v-BERT** [31] on the one hand performs contrastive learning discretizing continuous speech signals into a finite set of discriminative speech tokens. On the other hand, the model learns contextualized speech representations by solving a masked prediction task with the discretized tokens as input. During pre-training both tasks are simultaneously optimized in an end-to-end fashion. During fine-tuning the output of the pre-trained w2v-BERT model with 1B parameters is aggregated by a

LSTM decoder. On the Librispeech benchmark it has a similar WER of 1.4% as the leading system and on the Librispeech benchmark test-other the model achieves a SOTA of 2.5% WER. In addition, the model with 600M parameters was fine-tuned on a voice search task that allows users to use Google Search by speaking on a mobile phone or computer. It consists of voice snippets with an average duration of 5.5sec. The model was able to decrease errors by about 30% to 6.2. **SpeechStew** [21] uses the Conformer 1B with wav2vec pre-training. It is pre-trained on 7 available speech recognition datasets without any domain-dependent re-balancing or re-weighting. Without a language model it achieves a WER of 1.7% on LibriSpeech.

**TERA** [98] is a self-supervised speech model using a multi-target auxiliary task to pre-train a transformer encoder on a large training set of unlabeled speech. The input can be any acoustic features, such as MFCC. The model learns by reconstructing acoustic frames from modified samples which were randomly changed with respect to three properties: Time alteration requires the reconstruction from corrupted blocks of time steps. Channel alteration has to restore the signal from missing blocks of frequency channels. Magnitude alteration involves the regeneration of altered feature magnitudes. By reconstructing these data changes, the model learns a better contextualized representation. The time alteration width is set to 85 ms of speech, which is about the average phoneme duration. The largest model similar to BERT has 170M parameters. The model has strong results for phone classification, speaker recognition, and speech recognition, e.g. on the TIMIT benchmark with 14.5% phone error rate (PER).

In a comprehensive analysis, Zhang et al. [195] evaluate the benefit of self-supervised pre-training for ASR. They employ Conformer models with 600M to 8B parameters pre-trained and self-trained on extremely large and diverse unlabeled datasets containing thousands to a million hours of audio (*BigSSL*). Using only 3% of the labeled data they obtain comparable results to the SOTA of the Voice Search benchmark. On eight ASR benchmarks they are able to match or improve SOTA after pre-training. On five non-ASR task such as language identification and emotion detection, they can improve SOTA. For large datasets, the gains from pre-training are smaller but still significant.

Many applications benefit from understanding not only words but also other information, such as a person's emotion during an utterance, whether the speaker is wearing a mask, or whether the speech is synthetic. Shor [156] presents a large-scale, conformer-based architecture with more than 600M parameters that can be fine-tuned to detect these additional features and delivers SOTA performance.

### Available Implementations

- Conformer: https://github.com/PaddlePaddle/PaddleSpeech
- wav2vec: https://github.com/facebookresearch/fairseq sequence modeling toolkit for translation, summarization, language modeling and other text generation tasks.
- Tera: https://github.com/s3prl/s3prl

- Hugging Face speech recognition: https://huggingface.co/models?pipeline_tag=automatic-speech-recognition
- TensorFlow SST: https://tfhub.dev/s?module-type=audio-stt

### *7.1.4 Text-to-Speech*

Speech synthesis is about generating speech from another modality like text, lip movements, etc. A *Text-to-Speech* (*TTS*) system aims to convert natural language text into speech. *Mean Opinion Score* (*MOS*) is the most frequently used method to evaluate the quality of the generated speech. MOS is defined as the arithmetic mean over single ratings performed by human raters for a given stimulus in a subjective quality evaluation test. MOS has a range from 0 to 5, where real human speech is between 4.5 and 4.8. A comprehensive and up-to-date survey of TTS systems is provided by Tan et al. [163].

While earlier TTS systems simply concatenated prerecorded speech segments, modern systems perform a complete synthesis of speech. **WaveNet** [114] was the first model that successfully modeled the raw waveform of the audio signal instead of the acoustic features. It is able to generate new speech-like waveforms at 16,000 samples per second. WaveNet in its core is an autoregressive model consisting of dilated convolutions where each sample depends on the previous ones. In each layer the number of included time steps is doubled. WaveNet was able to increase the MOS-value from 3.86 to 4.21. *Fast WaveNet* was able to reduce the quadratic time complexity to linear complexity by caching previous calculations.

**Tacotron 2** is a neural network architecture for speech synthesis directly from text. It consists of a recurrent LSTM sequence-to-sequence feature prediction network with attention, which predicts a sequence of mel spectrogram frames from an input character sequence and a modified version of WaveNet, which generates time-domain waveform samples conditioned on the predicted mel spectrogram frames. Tacotron 2 achieved an impressive MOS of 4.53.

As TTS performs sequence processing similar to NLP, it is only natural that PLMs are also used in this area. Transformer-based models aim to mitigate two problems of previous TTS methods such as Tacotron 2: their high computational cost for training and inference, and the difficulty of modeling long dependencies with LSTMs.

**Transformer TTS** [94] adapts the original transformer encoder-decoder [168] to speech synthesis. The encoder receives phonemes as input, which are adapted by an encoder pre-net consisting of a CNN and a fully connected layer. The standard transformer encoder outputs contextual phoneme embeddings (Fig. 7.2). The decoder receives mel frames as input, which are converted by a decoder pre-net with two fully connected layers to generate appropriate embeddings. The standard decoder generates mel frames output embeddings. These are further processed by two different linear projections to predict the mel spectrogram and the stop token respectively. A 5-layer CNN produces a residual to refine the reconstruction of mel

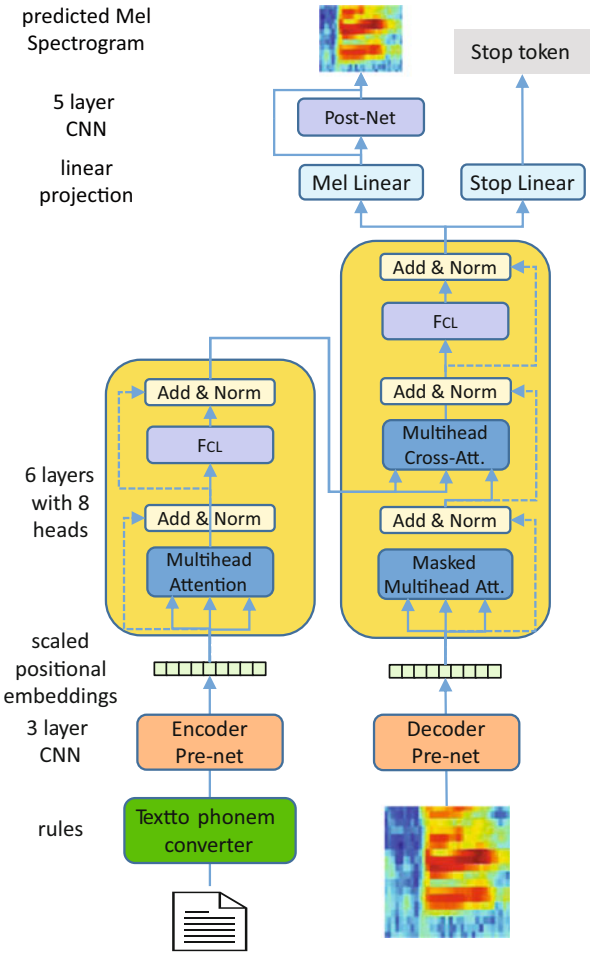


**Fig. 7.2** Speech synthesis with the transformer TTS. The encoder as well as the decoder have 6 layers with 8 attention heads and residual connections. The resulting mel spectrogram is transformed into the final audio output by a WaveNet vocoder [94]. Image credits in Table A.3

spectrogram. A WaveNet vocoder generates the final audio output. Both the encoder and decoder of the Transformer consists of 6 layers with 8 heads. The model is about 4.25 times faster than Tacotron 2 and achieves a MOS of 4.39 close to human quality.

**FastSpeech 2** [138] tackles the problem that an input text can correspond to multiple possible speech sequences due to variations in speech, such as pitch, duration, sound volume and prosody. It encodes the input phonemes by a transformer encoder to generate embeddings. Then a variance adaptor adds different variance information such as duration, pitch and energy into the hidden sequence. Finally,

the mel-spectrogram decoder converts the adapted hidden sequence into mel-spectrogram sequence in parallel. Both the encoder as well as the mel-spectrogram decoder have layers containing transformer blocks and 1D-convolutions. The variance adaptor predicts not only the duration, but also pitch and energy, using layers with 1D convolutions, feedforward layers, and layer normalization with dropout for regularization.

The variant *Fastspeech 2s* directly generates waveform from text without cascaded mel-spectrogram generation (acoustic model) and waveform generation (for example a vocoder, like wav2vec). The final waveform decoder consist of gated activations as well as different types of 1d-convolutions and dilated 1d-convolutions to cover a wider time range. The authors employ adversarial training in the waveform decoder to force it to implicitly recover the phase information by itself.

In their experiments the authors determine the following MOS-values: Tacotron 2: 3.70, Transformer TTS: 3.72, FastSpeech 2: 3.83, FastSpeech 2s: 3.71, and human speech: 4.30. Note that the difference to human speech is mainly caused by the vocoder. In addition, FastSpeech 2 and FastSpeech 2s are about 50 times faster than Transformer TTS at inference time.

**AdaSpeech 2** [186] adapts a TTS system to a target speaker. Only sound recordings of the target speaker without text transcription are required. The authors apply a mel-spectrogram encoder to a well-trained TTS model to conduct speech reconstruction, and at the same time constrain the output sequence of the mel-spectrogram encoder to be close to that of the original phoneme encoder. The mel encoder also consists of 4 feed-forward Transformer blocks. Note that the original system does not need to be retrained, only the mel encoder. During the fine-tuning to the target speaker, the mel decoder parameters are adapted. The model achieves on-par MOS voice quality with the transcribed TTS adaptation.

Recently Amazon has announced that Alexa will be able to mimic the voices of other persons [17]. To “make memories last” Alexa could, for instance, tell stories and play music using the voice of the deceased grandmother. Amazon notes, that it would take only about a minute of audio recording to imitate a voice.

### Available Implementations

- Tacotron 2: https://github.com/NVIDIA/tacotron2
- TransformerTTS: https://github.com/as-ideas/TransformerTTS
- FastSpeech 2: https://github.com/ming024/FastSpeech2
- AdaSpeech 2: https://github.com/rishikksh20/AdaSpeech2
- Hugging Face TTS: https://huggingface.co/models?pipeline_tag=text-to-speech
- Mozilla TTS Text-to-Speech for all: https://github.com/mozilla/TTS
- TensorFlow TTS: https://tfhub.dev/s?module-type=audio-speech-synthesis

### 7.1.5 Speech-to-Speech Language Model

**GSLM** [89] is a language model which receives raw speech audio as input and directly generate outputs. It can, for instance, be used to create a dialog system without intermediate text representation. Internally the model converts incoming raw speech to discrete pseudo-text units. As discretizers CPC [113], wave2vec 2.0 [10], and HuBERT [68] were used to create embeddings of varying length (50, 100, 200). The selection of units is difficult, as there is no vocabulary of sound units, and sound units have variable length with no obvious segmentation. Similar to BERT, HuBERT is trained with a masked prediction task using masked continuous audio signals as inputs. In experiments HuBERT performed best in most cases, followed by CPC.

The autoregressive "unit-based" language model has 12 layers and is trained on samples with up to 3k units generated from the 6k hours *LibriLight speech data* [139]. To generate speech from units a modified version of the Tacotron-2 model [154] was employed, which takes pseudo-text units as input and outputs a log Mel spectrogram. To generate waveforms the pre-trained vocoder *WaveGlow* [125] was used, which converts the log Mel spectrogram to speech.

In a first test the speech input was encoded into units, which were translated to speech. Here the intelligibility of the resulting speech is assessed by a human MOS opinion score. When trained on the *LJ Speech data* [74] the unsupervised model achieved a MOS (Mean Opinion Score) score of 4.00, while the combination of an ASR and TTS system achieved a slightly better score of 4.04 [89]. When testing the full language model generation, the model achieved a MOS score of 4.01, while the combination of ASR and a language model yielded a score of 3.91. According to the authors, the generated speech sounds like English, has recognizable phonemes and words. Examples show that improvements are needed at the language and syntax level. For sound transcription 200 units were good, while for language modeling a smaller number of units seems to be better. It can be expected that the quality can be improved with additional training data.

### 7.1.6 Music Generation

Foundation Models can also be applied to other sequence data, e.g. music. On the one hand a music language model can be trained, which is able to generate new music corresponding to the training data. On the other hand, a model can generate music conditioned on external information, e.g. lyrics or video. Bilici [14] provide a survey on recent music generation models.

A prominent approach to music generation is **MuseNet** [123] which employs the Sparse Transformer, a variant of GPT-2. It calculates attention patterns over a context of 4096 MIDI characters. To generate new compositions, one can select a composer and use the starting notes of a known piece. Then up to ten different

instruments can be selected, and the system will generate a piece of music with the required characteristics. The ratings of experts are quite favorable. Similarly, the **Music Transformer** [71] generates piano pieces. **Theme Transformer** [155] receives a theme as input and is trained to include this theme multiple times in its generation result.

**Jukebox** [36] adopts a multiscale vector quantizer variational autoencoder model (VQ-VAE) [113] to compress raw audio to discrete codes. This is based on an autoregressive Transformer and works also for human voices. Three separate VQ-VAE models with different temporal resolutions are employed. The trained model can be conditioned on an artist and a genre to steer the musical and vocal style, and on unaligned lyrics to make the singing more controllable. The model is capable of generating pieces that are many minutes long, and with recognizable singing in natural-sounding voices. A number of samples are available [35].

**CMT** [38] generates background music for a specific video. It aims to match the rhythm, timing, and movement speed of the video. CMT extracts these features from the video and allows global control of the music genre and instruments. The model does not require paired video and music training data. Experiments demonstrate that the generated background music has achieved satisfactory compatibility with the input videos, and at the same time, impressive music quality.

**Available Implementations**

- CMT Controllable Music Transformer https://github.com/wzk1015/video-bgm-generation
- Jukebox: A Generative Model for Music https://github.com/openai/jukebox

### *7.1.7 Summary*

Speech recognition has shown an enormous progress in recent years and Foundation Models are now an established approach to this task. They are combined with CNN blocks and are able to capture interactions over long distances and reduce processing times. Similar to NLP, self-supervised learning has led to great performance gains. Instead of tokens, as in NLP, discrete sound representations are generated. A number of different models follow this scheme, and they are able to increase SOTA on different benchmarks.

The generation of speech from text has improved dramatically in recent years. WaveNet was the first model to generate speech-like waveforms at 16,000 samples per second. Transformers can be used to convert input phonemes to mel spectrograms, from which a vocoder can generate speech audio. There are variants like FastSpeech 2s, which directly transform text to an audio signal. The output quality of the models is close to human speech. Some models are able to adapt their output to the voice of individual speakers. This is impressive, but also a major security

problem if in this way false utterances are produced imitating a person's voice. The recent S4 state-space model for long input sequences was able to reduce errors by 60% for classifying speech signals. It can be expected that this model will also lead to a considerable reduction of errors in other speech recognition tasks.

Speech recognition and text-to-speech can be integrated with other applications. SpeechBert [30] is an end-to-end Speech Question Answering (SQA) model by encoding audio and text with a single Transformer encoder, which is pre-trained with MLM on speech and text corpora and fine-tuned on Question Answering. Live speech translations are generated on-the-fly in a smartphone and allow a seamless communication in a foreign language [78, 81]. And GSLM is a generative language model, which directly processes discretized sound tokens.

Music generation is a related topic. Autoregressive PLMs, e.g. MuseNet or Music Transformer, can be used to generate music based on a pre-training with a large corpus. Here the composer style and the instrument may be selected. In addition, music can be conditioned on some input, e.g. lyric text for the Jukebox model or a video to compose background music.

## 7.2 Image Processing and Generation

The breakthrough of Foundation Models in NLP has generated tremendous interest in the computer vision community to adapt these models for vision and multi-modal learning tasks. Two factors are important for their success: self-attention and self-supervision. Self-attention layers generate representations that take into account the relationships between the tokens (text token and/or visual tokens). Self-supervision predicts masked or modified parts of data elements during training in large-scale datasets. It allows gaining enormous knowledge about the data without manually annotating it and assumes minimal inductive biases compared to other models like CNN and RNN. Comprehensive surveys on Foundation Models for vision and language applications are provided by Khan et al. [84] and Du et al. [43]. Hafiz et al. [62] give an overview over attention mechanisms and Deep Learning for machine vision. There is a recent tutorial on vision and language research [6]. The main features of the models discussed in this section are compiled in Table 7.2.

### *7.2.1 Basics of Image Processing*

Image processing can solve a variety of tasks, as shown in Fig. 7.3. The main content of an image can be described by classifying the most important object in the image. More demanding is the identification and classification of relevant objects in an image. This also requires the description of the object positions by bounding boxes. Creating a caption for an image involves identifying the most important objects in the image, how they relate to each other, and describing them using a natural

**Table 7.2** Main techniques to combine text and images.**Benchmarks:** VQA: COCO Visual Question Answering dataset (Sect. 7.2.5) [56]; img-gen: MS-COCO image generation benchmark with fine-tuning; img-gen-0: MS-COCO image generation benchmark zero-shot; ImageNet: ImageNet classification top1 accuracy; captions: MS-COCO image captioning benchmark; FID: Fréchet Inception Distance should be small (Sect. 7.2.6) [64]. Numbers in parentheses are parameter counts

| Model | Approach | Benchmark |
|---|---|---|
| Vision Transformer (ViT) Sect. 7.2.2 | Concatenate text tokens and image token generated from image patches. Process with a BERT autoencoder and perform classification (632M) | ImageNet SOTA acc. 90.5% |
| CLIP Sect. 7.2.4 | Encode image with vision transformer and text with a GPT autoencoder. Maximize similarity of image and text embeddings, predict if they belong together | |
| VilBERT Sect. 7.2.5 | Extract bounding boxes with Faster R-CNN. Image regions and text are encoded by two BERT autoencoders and perform cross-attention. Fine-tuned to VQA | VQA SOTA 70.9% |
| OSCAR Sect. 7.2.5 | Extract bounding boxes with Faster R-CNN. A BERT autoencoder associates region descriptions with text. Fine-tuned for 7 tasks, e.g. image captioning | captions SOTA 41.7 BLEU-4 |
| VinVL Sect. 7.2.5 | Uses ResNeXT model as region extractor and OSCAR. Fine-tuned for image captioning | captions 40.4 BLEU-4 |
| DALL-E Sect. 7.2.6 | Text is encoded as tokens, image is transformed to image tokens by variational autoencoders (VAE). Uses GPT-3 (12B) to generate new image tokens | img-gen-0 17.9 FID |
| GLIDE Sect. 7.2.7 | Reverses diffusion which destroys an image. Generates image by small changes with U-Net model (3.8B) | img-gen-0 SOTA 12.2 FID |
| XMC-GAN Sect. 7.2.7 | GAN-based image generator, generator creates images, discriminator discriminates fake and real images | img-gen SOTA 9.3 FID |
| CogView Sect. 7.2.7 | Vector quantized VAE. GPT-model (4B) is trained with text tokens and quantized image tokens | img-gen SOTA on blurred images |
| LAFITE Sect. 7.2.7 | Uses CLIP to transform text to image embeddings. Train to modulate layers of StyleGAN2 [82] to generate images | img-gen SOTA 8.1 FID<br>img-gen-0 16.9 FID |

(continued)

**Table 7.2** (continued)

| Model | Approach | Benchmark |
|---|---|---|
| OFA Sect. 7.2.8 | Uses text, image tokens and objects with bounding boxes. Seq2seq model (472M) pre-trained to associate tokens and objects. Text instructions control 9 different tasks | img-gen SOTA 10.5 FID captions SOTA 43.5 BLEU-4 |
| DALL-E 2 Sect. 7.2.7 | Generate in image embedding from text by CLIP, transform to 1024 × 1024 image by diffusion decoder | img-gen-0 SOTA 10.4 FID |
| Imagen Sect. 7.2.7 | Generate text embeddings by T5-XXL, generate image patches by diffusion model, upsampling to 1024 × 1024 by two superresolution diffusion models | img-gen-0 SOTA 7.3 FID |
| Stable Diffusion Sect. 7.2.7 | Generate images using U-Net and diffusion | ImageNet conditional 3.6 FID |

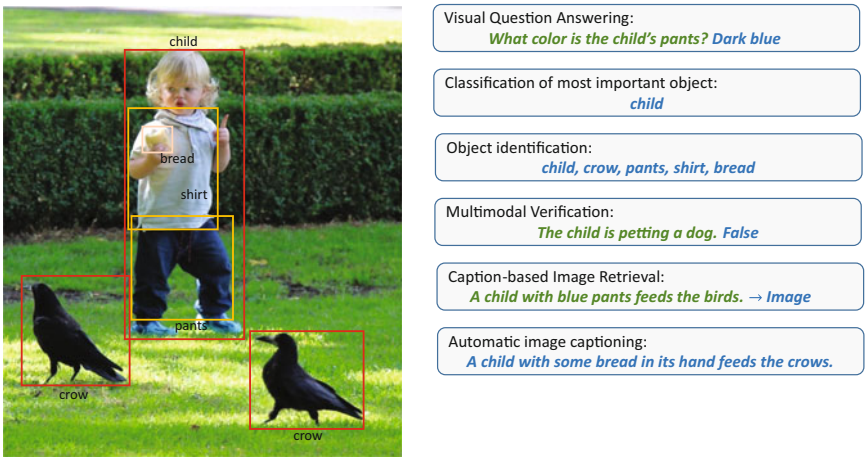


**Fig. 7.3** Image analysis can be used to solve a number of different tasks. Depending on the task, the system receives a text (green) and an image as input and generates a text (blue) and an image as output. Image credits in Table A.3

language sentence. Related to this is the retrieval of an image that corresponds to a caption. Visual question answering requires interpreting a question and analyzing the image to generate an answer in natural language. A variant is multimodal verification, where the truth of a statement about the image has to be assessed.

Many tasks involve the creation of a new image. A prominent example is the generation of a completely new image according to a caption. Alternatively a missing image area can be filled in. A variant is to change the style of an image according to a caption, e.g. from a photo to a painting in the style of van Gogh. This can be also performed for a specific image region.

An important aspect is the representation of images for transformers. Language models partition text into a sequence of tokens, which form the input of a transformer. The same approach is chosen for images, which are partitioned into small image patches. The contents of each patch can be represented by a vector, which forms the input of the transformer. The location of the patch is encoded by a position embedding, which is added to the input embedding.

The embedding of an image patch can be simply a learnable linear transformation of its pixel values. Other transformations may be used, e.g. small CNN models or variational autoencoders (Sect. 1.7). To get more robust representations, the generated vectors are often discretized to get rid of local noise. In addition, text from a caption or region annotation can be used as input. As usual, this text is converted to tokens from a vocabulary.

To model the interaction between image elements and text, different transformer architectures can be used (Table 7.2). A *single stream architecture* concatenates all inputs and processes them with a single transformer. This allows to determine interactions between different input elements, but requires the handling of long sequences. Dual-stream or *multi-stream architectures* process different modalities or image resolutions by separate PLMs. In this case the input sequences are shorter. Various forms of interaction between the streams have been proposed (e.g. cross-attention). Later the outputs may be compared by similarity measures or combined by other PLMs.

The pre-training task for vision follows the pattern of the text transformer. *Masked language modeling* (MLM) masks a fraction of the input tokens and requires the model to predict the tokens from the context. If there are text and image tokens, the information in both modalities can be utilized for this task and the model learns the association between text and image elements. Similarly, image regions can be masked and reconstructed from the text and image context. In a classification task, the model can determine whether a caption correctly describes an image or is some random text. In this way, the correlation between text and images can be trained. Another goal is to learn a joint image and word representation in the same semantic space by pushing together the embeddings of matched image-text pairs, while pushing apart the non-matched pairs. For this image-to-text *contrastive loss*, the proximity of embeddings is measured by a scalar product between the embeddings.

### 7.2.2 Vision Transformer

The **ViT** (Vision Transformer) [42] applies a pure Transformer encoder (Sect. 2.3.1) to image patches. The input image $\boldsymbol{x} \in \mathbb{R}^{H \times W \times c}$ has $H \times W$ pixels and $c$ color channels. It is partitioned into patches of $s \times s$ pixel, e.g. $s = 16$. Each of the $N = HW/s^2$ patches consist of $s^2 * c$ numbers, which are linearly mapped to a vector of length $d$ used as the inputs of the transformer. Usually, a one-dimensional position embedding is added, because two-dimensional positions gave no significant

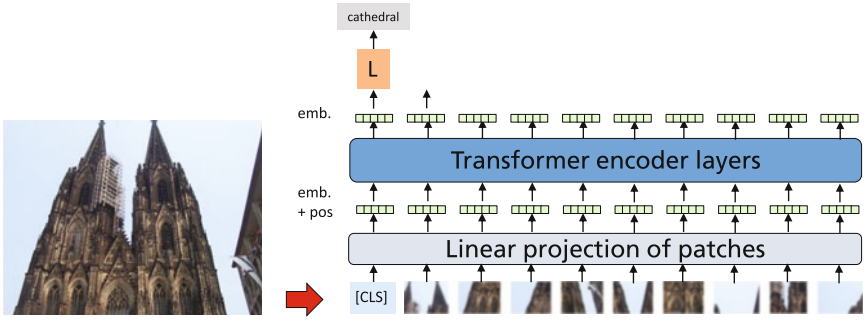


**Fig. 7.4** The Vision Transformer ViT partitions an image into square patches of fixed size. For each patch an embedding is calculated by a linear projection. A standard encoder computes contextual embeddings. The embeddings of the [CLS] token is used to compute a class by a logistic classifier [42]. Image adapted from [42] with permission of the authors, credits in Table A.3

performance improvement. Different models $\text{ViT}_{\text{Base}}$, $\text{ViT}_{\text{Large}}$, and $\text{ViT}_{\text{Huge}}$ with 12, 24, and 32 layers and 86M, 307M and 632M parameters respectively are employed.

The transformer encoder has an input sequence length of $N$ consisting of vectors of size $d$. Each layer generates $N$ embeddings of length $d$. The output embedding of the [CLS] token in the last encoder block is the input to a logistic classifier to compute probabilities of the image classes. The architecture is shown in Fig. 7.4.

It is remarkable that the images may be trained with varying input image resolutions. But patch size is always the same yielding different input size lengths. To take the new resolution into account, a 2D interpolation of the position embeddings is performed. The model is typically pre-trained on a large dataset JFT-300M [161] to predict masked inputs. It is fine-tuned with a smaller task using a different classifier layer. It is often beneficial to fine-tune at higher resolution than pre-training [189]. The models were pre-trained on datasets with up to 300M images.

The largest model $\text{ViT}_{\text{Huge}}$ has input patches of size $14 \times 14$. It was able to outperform an improved and pre-trained ResNet152 [63] with 152 CNN layers and EfficientNet [92] on ImageNet, and achieved a SOTA of 90.5% Top-1 accuracy for the classification of images into 1000 object categories [118]. Pre-training increases absolute accuracy by 13% on the test set of ImageNet. With 2.5k TPUv3 days, it required only 25% of the computing effort (including pre-training) required for ResNet. It improved SOTA for another 5 popular image classification benchmarks. The smaller $\text{ViT}_{\text{Large}}$ with input patches of size $16 \times 16$ also outperformed ResNet152 requiring only 6.8% of ResNet152's compute effort.

When ViT is trained on a moderate dataset like ImageNet, the model achieves a performance below that of ResNet (Sect. 1.7) with a comparable parameter count. It seems that CNNs have more appropriate inductive biases, such as translation equivariance and locality, which the transformer must learn through pre-training. Therefore, only pre-trained transformers can outperform CNNs, but this requires a lower computational effort. Cao et al. [20] present a method how ViTs can be trained

with limited data and achieve good results. Chefer et al. [22] present a new method based on Taylor decomposition methods to visualize the parts of the image that led to a certain image classification.

It is instructive to analyze the inner structure of a trained model. It turns out that the trained position embeddings reflect the row and column structure of the input image, and patches in the same row/column have similar embeddings. Based on the attention weights, it can be determined which image parts are considered by a specific attention head. Some attention heads take into account the whole image while others have consistently small attention distances in the lower layers. This could have a similar function as early convolutional layers in CNNs [130]. An experimental investigation has shown that transformers are highly robust to severe occlusions [108]. In contrast to CNNs, which often detect an object based on texture and less on shape, ViTs are comparable to humans on shape recognition. Figure 7.5 shows attention regions for the whole ViT model corresponding to semantically relevant areas.

A number of researchers have investigated the robustness of ViT. In a series of experiments, Mao et al. [103] found that the ViT tends to employ local features containing textures and noise, and to some extend ignores global context such as shape and structure. In response, they propose to discretize the continuous input features to image tokens using a vector quantizer based on a variational autoencoder (*VQ-VAE*) [113]. They report accuracy improvements of up to 12% on

**Fig. 7.5** The input image is shown in the upper row. The lower row depicts the area of main attention computed by the Vision Transformer model to the input space for classification. Image reprinted with kind permission of the authors [42, p. 8]

several ImageNet classification benchmarks. A similar adaptive token generation methods for the ViT was proposed by Ryoo et al. [146]. **BEiT** [11] outperforms, the supervised pre-trained ViT using a self-supervised method inspired by BERT (masked image modeling) and based on a VQ-VAE.

### *7.2.3 Image Generation*

There are a number of Foundation Models for various image enhancement tasks. Image *super-resolution* converts a low-resolution image to a higher resolution. **SwinIR** [96] is based on a hierarchical representation starting from small-sized image patches and gradually merging neighboring image patches in deeper layers. For training, the model gets a small-scale image as input, which is preprocessed with a CNN layer. The transformer block contains transformer and CNN layers and is trained to reconstruct the high-resolution image. SwinIR achieves SOTA on benchmarks for super-resolution, image denoising, and JPEG compression artifact resolution, while having only 12M parameters.

**ColTran** [88] transforms a grayscale image to a fully colored image by using transformers with column and row attention. It first predicts colors by a conditional transformer for a spatially reduced image with only 512 coarse colors. Two subsequent fully parallel transformers upsample the coarse colored low resolution image into a fully colored high resolution image. The model achieves the best FID-score (Sect. 7.2.6) of 19.7 on ImageNet data compared to different alternatives. Examples of colorizations are shown in Fig. 7.6.

**Fig. 7.6** Different colorizations of grayscale images (left) by ColTRan [88]. Note that semantic constraints, e.g. the color of the skin and the tree leaves, are usually respected. Image reprinted with kind permission of the authors [88, p. 1]

**Fig. 7.7** VQ-GAN [45] enables transformers to synthesize high-resolution images with 1280×460 pixels. Image reprinted with kind permission of the authors [45, p. 12873]

The **Swin Transformer** [99] constructs a hierarchical representation of an image by starting from small-sized image patches and gradually merging neighboring patches in deeper Transformer layers. A linear computational complexity is achieved by computing self-attention locally within non-overlapping windows of size 7 that partition an image. Between consecutive layers the attention windows are shifted such that there is an overlay with the neighboring windows of the prior self-attention layer. The largest model version has 197M parameters and processes images of resolution 384 × 384. On ImageNet classification the model achieves a top-1 accuracy of 87.3%. Also on object detection in images, the Swin Transformer is able to improve the prior best results.

**VQ-GAN** [45] uses a CNN to efficiently learn a codebook of context-rich visual patches, and subsequently learns a model of their global structure. The long-range interactions within these patches require an expressive GPT-2 to model distributions of the visual patches. The dictionary of image patches captures perceptually important local structure according to *perceptual loss* [41, 194]. This loss is optimized with an adversarial training procedure with a patch-based image discriminator that aims to differentiate between real and reconstructed images.

A GPT-2 model with 307M parameters is pre-trained to generate the code sequence of encoded images in an image corpus. Each image is partitioned to 16×16 patches with a sequence length of 1024. An example image is shown in Fig. 7.7. If the training corpus contains class information $c$, images of specific classes can be generated. Class information can also be restricted to specific image regions. While VQ-VAE yields an FID of about 10 for the reconstruction of ImageNet photos, VQ-GAN achieves a much better value of 1.7.

**StyleSwin** [191] is a further development of VQ-GAN. It uses the *Swin transformer* [99] discussed above. StyleSwin employs a wavelet discriminator in the spectral domain to suppress blocking artifacts. The model with 41M parameters achieves SOTA quality on multiple established benchmarks. Example images are shown in Fig. 7.8 having a coherent global geometry and high-fidelity details. On the CelebA-HQ 1024 benchmark StyleSwin yields an FID of 4.4, which is better

**Fig. 7.8** Images in the 1024×1024 resolution generated by StyleSwin [191] on FFHQ 1024×1024 data (left) and CelebA-HQ 1024 × 1024 data (right). Best seen with zoom. Image reprinted with kind permission of the authors [191, p. 8]

than all prior models including StyleGAN2 [82]. For the task of generating churches based on the LSUN dataset StyleSwin has an FID-score of 3.1, which is nearly as good as the best scoring adversarial CIPS model [7] with an FID-score of 2.9.

**Data2vec** [9] proposes a new training criterion for self-supervised learning, which can be applied to image, text and speech data. It has two kinds of models: a teacher model, which processes the whole input, and a student model, which processes the input while masking some data.

The model employs a standard transformer architecture with media-specific input encoding. Images are encoded by linearly transformed image patches similar to ViT. Speech data is encoded by multi-layer 1-D convolutions. Text data is encoded as subword tokens. Training targets for the student model are constructed from the averaged top $K$ encoder blocks of the teacher network, which processes the complete input. This target has to be predicted by the student model, which only receives the masked inputs. Representations of data2vec are continuous and contextualized through the use of self-attention, which makes them richer than a discrete set of tokens used for other approaches.

Separate models are trained according to this scheme for speech, images and text. For images a Data2vec model achieves a new SOTA of 86.2% top-1 accuracy on ImageNet-1k with restricted training set. For speech data, the model reaches a WER of 5.5% on the Librispeech test-other benchmark. For language processing, Data2vec has an average score of 82.9 on GLUE, which is better than RoBERTa. This demonstrates that the model can be effective for multiple modalities. It can be expected that this model will be extended to learn across modalities.

### *7.2.4 Joint Processing of Text and Images*

Once transformers were applied to text and images, joint processing of both modalities became an obvious alternative. Three steps are required for this:

- encoding images and texts into embeddings preserving their semantics;


**Fig. 7.9** MS-COCO dataset [26]: images similar to sample images from the dataset. The corresponding captions indicate the level of detail. Image credits in Table A.3

- designing powerful architectures to model the interaction between both modalities;
- developing effective pre-training tasks.

After learning universal vision and language features, these PLMs can be fine-tuned on various downstream vision-language tasks.

For pre-training large scale datasets of text-image pairs $(\boldsymbol{v}, \boldsymbol{u})$ are required. Each pair consists of a sequence $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_T$ of text tokens and a sequence $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_R$ of image features or *visual tokens*, e.g. image patches. In this way, we can unify input representation as sequence of embeddings for both modalities. An example dataset is *COCO captions* [26], which contains 328k images of 91 object types of common objects in their natural context together with the corresponding image captions (Fig. 7.9). Other datasets like *Conceptual Captions* (CC) [153], *RedCaps* [34], and *Laion* [151] contain 3.1M, 12M and 400M images respectively together with captions or descriptive text.

Pre-training tasks have to be designed in such a way that the model has to reconstruct parts of the text or image based on the remaining contextual text and image features. For *Cross-modal MLM* (Masked Language Modeling) the model has to predict masked tokens or image patches based on the other unmasked text tokens and visual tokens. Here different masking strategies can be used such as whole word masking, masking text spans, or permuting tokens (Sect. 3.1). *Masked region prediction* aims to predict the content of an image region. Objects and their regions are annotated manually or by an auxiliary model. Then the model is required to predict the object (or a distribution over objects) for that region. In this way, the model learns to locate objects in an image.

**CLIP** [126, 127] is trained to predict a score indicating which image caption corresponds to which image. Given a batch $(\boldsymbol{v}_1, \boldsymbol{u}_1), \ldots, (\boldsymbol{v}_n, \boldsymbol{u}_n)$ of tokenized text-image pairs, CLIP has to predict which of the $n \times n$ possible $(\boldsymbol{v}_i, \boldsymbol{u}_j)$ pairings across the batch actually occurred. By *contrastive learning*, CLIP creates a multi-modal embedding space by jointly training an image encoder and text encoder to maximize the cosine similarity of the image and text embeddings of the $n$ real pairs in the batch while minimizing the cosine similarity of the embeddings of the $n^2 - n$ incorrect

pairings. This contrastive training with positive and negative examples has been shown to outperform alternatives. As image encoder a Vision Transformer (ViT) with images patches of size $14 \times 14$ (Sect. 7.2.2) was employed, which works better than a ResNet [63] encoder based on CNNs. Text was enclosed by [SOS] and [EOS] tokens and a 12 layer autoregressive GPT model was used to compute embeddings. The embedding of [EOS] in the highest layer was employed as the representation of the whole text.

CLIP was trained on 400M image-text pairs of the *WIT data* [127] to associate an image with the best-matching caption. In addition, the prediction of the next token was used as an auxiliary loss term for the GPT model. The model can be used to retrieve a text best fitting to an image, or an image optimally corresponding to a text.

The resulting model has acquired a comprehensive knowledge about text and images. With a top-1 classification accuracy of 76.2%, it even surpasses the top-1 classification accuracy of 75.0% of the original ResNet50 on ImageNet zero-shot classification without the need to use any of the 1.28M training examples that ResNet50 was trained on. Hence, CLIP can be considered a 'zero-shot classifier'. This also holds for 16 out of 27 other image classification benchmarks. When a linear classifier is fitted on top of CLIP's features, it improves CLIP's accuracy on the ImageNet test set by almost 10% [126]. If the image distribution is changed, e.g. to sketches, CLIP-based classifiers are much more robust. Zero-shot CLIP classifiers improve effective robustness by a large amount, especially with respect to distribution shift. This demonstrates that the inclusion of caption text into vision models enhances performance and robustness.

**BriVL** [46] is a similar model for Chinese language, which uses a larger set of negative examples stored in a queue. It uses a huge training dataset of 650M weakly correlated text-image pairs, where, for instance, an image of a birthday cake has the caption *"Happy birthday! Make a wish"*. It achieves SOTA results for cross-modal retrieval and visual question answering.

**ALIGN** [77] also uses separate encoders for text and images with a cosine-similarity combination function at the top. As image encoder an EfficientNet CNN is employed. BERT is trained to produce a text embedding for the [CLS] token. Again the similarity is minimized for genuine image-text pairs and maximized for random pairs. ALIGN has 675M parameters and uses a huge training set of 1.8B noisy image pairs. In spite of the noisy data the model achieves a slightly better accuracy (85.5%) on ImageNet top-1 classification than CLIP.

### *7.2.5 Describing Images by Text*

The automatic generation of a natural language description of an image is also called *image annotation* or *image captioning*. The task is challenging, as it requires visual perception, recognition, and real-world knowledge, as well as the *grounding* of language expressions in the image space. *Symbol grounding* describes, how words acquire their meaning, e.g. by associating a word with an object in an image. Aside

from determining and extracting the important objects and details of an image, the model has to infer the semantic relationship of the objects and the scene (Fig. 7.9).

Current top models for describing images work in two stages:

- an *object detection* model is pre-trained to encode an image and the visual objects in the image to feature vectors,
- a crossmodal PLM is pre-trained to associate text and visual features and generate a caption for an image.

Similar to language translation, various metrics are used to evaluate the generated texts, e.g. BLEU or ROUGE (Sect. 2.3.3). Surveys of image captioning techniques are provided by Hossain et al. [67], Oluwasammi et al. [112], and Stefanini et al. [159].

**VilBERT** [100] aims to learn representations that can jointly model images and natural language. It extracts bounding boxes and their visual features using a pre-trained object detection network (Faster R-CNN [137]). These image region features as well as the text are input to two separate transformer encoders (two-stream architecture). Subsequently, transformer layers with cross-attention in both directions are applied to learn cross-modal relationships. VilBERT was pre-trained on Conceptual Captions data.

The model was fine-tuned and evaluated on different tasks. *Visual question answering* (*VQA*) answers natural language questions about images. VQA is treated as a multi-label classification task with 3129 possible answers. Final embeddings of the text and image parts are fed into a classifier to estimate class probabilities. On the COCO test set VilBERT achieved a new SOTA with an accuracy of 70.9%. *Caption-based image retrieval* is the task of identifying an image from a pool given a caption describing its content. The model was fine-tuned on a Flickr dataset and had a recall@1 of 58.2%, thus establishing a new SOTA.

**OSCAR** [95] has the strategy to connect the relevant objects in the image with the corresponding phrases in the caption text. The authors use self-attention to learn these alignments, which can be significantly improved by additional object tags detected in images as reference points. Oscar represents each input image-text pair as a Word-Tag-Image triple $(w; q; v)$, where $w$ is the sequence of words of the caption text, $q$ contains the words of the textual object tags detected in the image, and $v$ is the set of the corresponding region images. A CNN model (Faster R-CNN [137]) is used to discover the objects in $q$ as well as to the corresponding regions $v$. For pre-training the transformer encoder, part of the tokens in $(w; q; v)$ are masked, and the model learns to predict the masked tokens. In addition, sometimes the $q$-terms are changed randomly. The model has the additional task to identify these modifications. A small and a large model version are trained with a sequence length of 768 and 1024 using a public corpus of 6.5 million text-image pairs. The model is fine-tuned to generate the caption according to the sequence-to-sequence objective. The model achieves a new SOTA on COCO-captions with respect to BLEU-4 (41.7%), METEOR and ROUGE-L as well as for several other captioning benchmarks.

**VinVL** [193] is pre-trained on three text-image corpora with 2.5M images, and can generate visual features with a richer collection of visual objects and concepts.

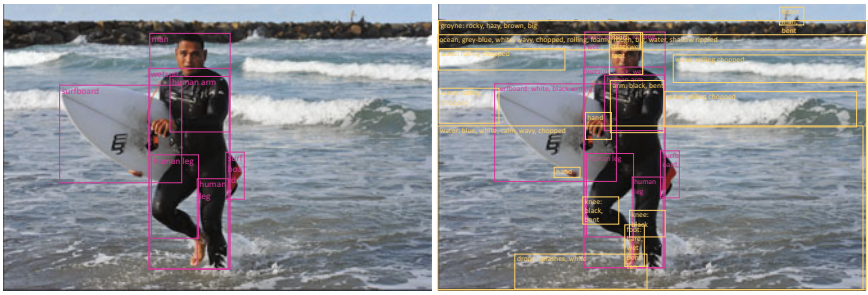

**Fig. 7.10** Standard bounding-box object descriptions (left) and detailed annotations, which can be generated by VinVL (right) and contain visual concepts and attribute information [193]. Image credits in Table A.3

VinVL pre-trains a large-scale object-attribute detection model based on the CNN-based ResNeXt-152 C4 architecture [179]. The model does not describe objects by a single noun, but by a large number of attributes and details, which enhances the performance in joint image-language tasks (Fig. 7.10). The approach is combined with OSCAR and yields an improved SOTA on image captioning. **VIVO** [70] is a similar transformer model trained to label image regions with 6.4k different object tags. VIVO is fine-tuned with COCO image-caption pairs and learns to generate caption sentences, also using object tags not appearing in the caption data. This is possible as VIVO can exploit large amounts of paired image-tag data to learn rich descriptions for images. On the test set VIVO generates better captions than humans according to the *CIDEr metric* [69], which counts the common words weighted by tf-idf in the generated and the reference text [169].

**SimVLM** [171] is a transformer encoder-decoder, which uses the first three blocks of ResNet to extract contextualized patches from images, and associates the image tokens with text tokens. The decoder then predicts the continuation of the textual sequence as shown in Fig. 7.11. It is trained on 1.8B noisy image text pairs and 800GB text documents. SimVLM achieves a new SOTA for visual question answering on the *VQA v2 benchmark* [56] with 80.3% accuracy. In addition, it reaches SOTA for visual entailment, visual reasoning, and image captioning on COCO captions with respect to Meteor (33.7).

**Frozen** is a Foundation Model trained to associate text with images. It can be instructed by few-shot learning to answer question on an image [166]. The language model is a pre-trained autoregressive model with 7B parameters trained on the C4 dataset with 807GB text [129]. The vision encoder is based on NF-ResNet-50 [16] and provides an embedding vector characterizing the image. During training the image embedding is used as a prefix before the token embeddings of the generated text. Using the *conceptual captions* dataset the vision encoder is trained while freezing the language model. The training target is to generate a caption for the image.


**Fig. 7.11** The SimVLM encoder-decoder model receives an image (top) and a text (middle) as input and produces an output text (bottom) [171]. The image patches are encoded by the first layers of ResNet. Image reprinted with kind permission of the authors [171, p. 3]

During inference, several examples consisting of an image embedding and token embeddings are fed into the language model, which generates an answer. An example is to caption a microscope with *"This was invented by Zacharias Janssen."*, and a light bulb with *"This was invented by Thomas Edison."*. After five seeds and the input of an airplane together with *"This was invented by"* the model generates the output *"the Wright brothers"*. In this way, different categorizations of images can be defined on the fly. These samples demonstrate the ability to generate open-ended outputs that adapt to both images and text, and to make use of facts that it has learned during language-only pre-training. The model is a proof-of-concept and shows a way to generate few-shot models for image-text tasks.

### *7.2.6 Generating Images from Text*

By training on text-image pairs, transformers can acquire the knowledge to generate images corresponding to text descriptions. By successively producing the next token with a language model, it is possible to predict visual tokens, which then can be synthesized to images. However, there are other image generation techniques.

- *Variational Auto-Encoders* (*VAE*) compress an input image to a small latent representation and reconstruct the image as good as possible. An additional loss term ensures that the distribution of latent representations follows a Gaussian [79].
- *Generative Adversarial Networks* (*GAN*) use a generator to transform a noise vector $\boldsymbol{s}$ to an image $\tilde{\boldsymbol{x}} = G(\boldsymbol{s})$. Then a discriminator $D(\boldsymbol{x})$ has the task to classify its input as synthetic image $\tilde{\boldsymbol{x}}$ or real image $\boldsymbol{x}$ [53]. Both networks are trained alternately with an adversarial loss.

Lee et al. [91] give a survey of techniques for text driven image generation and manipulation.

There are a number of approaches to measure the quality of generated images. The *Inception Score* (*IS*) [150] applies a CNN-based *Inception model* [162] trained on ImageNet to every generated image to get a conditional class label distribution, which should concentrate on few classes, i.e. have low entropy. In addition, many different classes should be generated for the test data, which is captured by the defined IS measure. The *Fréchet Inception Distance* (*FID*) [64] is an improved measure using the Fréchet distance between ImageNet classifier distributions, which measures the similarity of the distributions taking into account the location and ordering of the points along the graph. *CLIP Similarity Score* (CLIPSIM) [72] is based on the CLIP model (Sect. 7.2.4). It generates image and text embeddings with CLIP and calculates their cosine similarity.

**DALL-E** [133] uses a *GPT-3* autoregressive language model with 12B parameters to generate a new image from a textual description. The caption text of the image is BPE-encoded into 256 tokens. Then each $256 \times 256$ image is compressed to a $32 \times 32$ grid of image tokens using a discrete variational autoencoder. Each image token represents its $8 \times 8$ pixels by 8192 possible values. The caption tokens are concatenated with the $32 \times 32 = 1024$ image tokens forming the input sequence of GPT-3.

In the first stage the image tokens are trained yielding continuous image values. Then the discrete image tokens are obtained by training with a *Gumbel-softmax relaxation* [75] (Sect. 7.1.3). In the second stage a *Sparse Transformer* [27] with 64 self-attention layers and 12B parameters is trained to sequentially generate the joint input sequence. For the image tokens, special attention masks are used: row, column, or convolutional attention masks. The model was trained on 250M text-image pairs from the Internet.

For image generation, the authors rerank the samples drawn from the transformer using a pre-trained contrastive model, which assigns a score based on how well the image matches the caption. Figure 7.12 shows different images sampled from the algorithm. In a comparison to the prior model DF-GAN [165], the images generated by DALL-E were chosen as most realistic and more matching the caption in more than 90% of the time. Similarly, the images generated by X-LXMERT [28] look inferior.

**GauGAN2** [122, 149] combines segmentation mapping, inpainting and text-to-image generation in a single model. It is one of the first semantic image synthesis models that can produce photorealistic outputs for diverse scenes including indoor, outdoor, landscape, and street scenes. The recent version also can generate images according to text input. The model behind GauGAN2 was trained on 10 million high-quality landscape images. Details of the model are not known.

**XMC-GAN** [192] is a GAN-based text-to-image generation model containing a generator for synthesizing images, and a discriminator that is trained to discriminate real and generated images. It maximizes the mutual information between the corresponding pairs: (1) images (real or generated) with a sentence describing the scene; (2) a generated image and a real image with the same description; and (3) regions of an image (real or generated) and words or phrases associated with them.

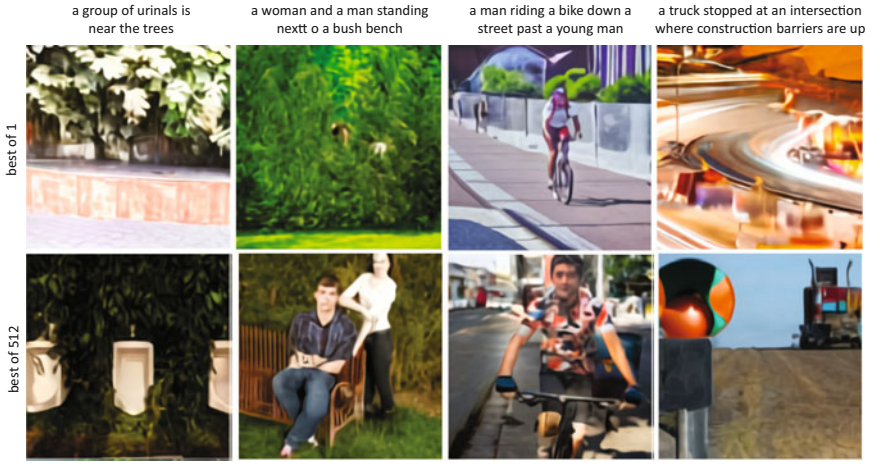


**Fig. 7.12** According to a natural language caption (top) a number of images are generated by DALL-E [133]. The middle row shows images generated by DALL-E corresponding to the caption. The lower row shows the best image from a sample of 512 automatically selected by a quality score. Image reprinted with kind permission of the authors [133, p. 6]

The goal is for the matching pairs (both text-to-image and real image-to-generated image) to have high similarity scores and for non-matching pairs to have low scores.

For the input text the model computes a global sentence embedding $emb_s$ and the word embeddings $emb_w$ from a pre-trained BERT module. $emb_s$ and random noise $z$ from a standard Gaussian distribution are concatenated to form the *global condition*, which is passed through several up-sampling blocks to generate a $16 \times 16$ feature map. The global condition is also used as the condition to calculate scale parameter and shift parameter in conditional batch normalization layers. The word embeddings $emb_w$ are input for an "attentional self-modulation layer" to generate fine-grained image regions. On MS-COCO, XMC-GAN improves the SOTA FID-score (Sect. 7.2.6) from 24.7 to 9.3, and is significantly preferred by human evaluators. Similarly, human raters prefer the image quality of XMC-GAN generated images 77% of the time, and 74% prefer its image-text alignment compared to three other SOTA approaches (CP-GAN, SD-GAN, and OP-GAN).

**Cogview** [40] employs a *Vector Quantized Variational AutoEncoder* (*VQ-VAE*). In the first stage, a discrete autoencoder is used to transform the image into a discrete sequence of tokens. In the second stage a GPT model learns to generate image tokens based on a prompt of SentencePiece text tokens. To generate image tokens, an encoder maps an image $\boldsymbol{x} \in \mathbb{R}^{H \times W \times 3}$ to $h \times w$ image patches, which are quantized to a nearby embedding in a learnable set $\{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k\}$ of embedding vectors $\boldsymbol{u}_i \in \mathbb{R}^d$ [113]. The decoder maps the embeddings back to the image, and the embeddings are selected to minimize the difference between output and input image.


**Fig. 7.13** Images generated by CogView [40] controlled by the text input (top). The image style can be influenced by the input text. The best of a sample of 60 images is selected. Image reprinted with kind permission of the authors [40, p. 1]

The GPT-model of CogView has 48 layers with a hidden size of 2560, 40 attention heads and 4B parameters. The input to the model is of the form *"[ROI1] <text tokens> [BASE] [BOI1] <image tokens> [EOI1]"* and contains special tokens. The pre-training task is to predict tokens from left to right for 30M text-image pairs in English and Chinese. A sparse attention pattern similar to BigBird (Sect. 3.2.1) is used.

As shown in Fig. 7.13, CogView has a similar performance in image generation as DALL-E. It achieves the SOTA FID on the blurred MS COCO dataset, outperforming previous GAN-based models and DALL-E, although DALL-E has three times more parameters. When evaluated by humans, CogView was able to beat GAN-based models by a large margin. However, generation of images with CogView is rather slow, because each image is generated token-by-token. In addition, the quantization leads to some blurriness in the images.

**LAFITE** [200] is a model for generating images from text. Image generation is based on *StyleGAN2* [82], which creates various image attributes by modulating the weights of the convolution kernels [177]. LAFITE generates these modulating signals based on language input. It relies on the multimodal semantic space of the pre-trained CLIP model (Sect. 7.2.4) to produce an image embedding $emb(\boldsymbol{x})$ from a text $\boldsymbol{x}$, and therefore does not need extra text data. This image embedding is inserted into the image generation model similar to StyleGAN2 by a GAN architecture. On the MS-COCO benchmark, LAFITE achieves a zero-shot FID value of 26.9, which is better than the values of DALL-E (27.5) and CogView (27.1). When fine-tuned on MS-COCO, LAFITE has a FID-score of 8.1, which is better than that of XMC-GAN (9.3) and other GAN models. Note that LAFITE only has 75M trainable parameters.

### 7.2.7 *Diffusion Models Restore an Image Destructed by Noise*

**GLIDE** [109] is an image generation technique based on a *diffusion model*. A diffusion model describes the process of systematically and slowly destroying structure in a data distribution through an iterative forward *diffusion process*, e.g. the addition of noise [157]. To the data $\boldsymbol{x}^{[0]}$, e.g. a matrix of pixel values, we can apply

Gaussian diffusion distribution $q(\boldsymbol{x}^{[t]}|\boldsymbol{x}^{[t-1]})$, where a Gaussian with expectation $\mathbf{0}$ and covariance $\beta\boldsymbol{I}$ is added. This yields a series $\boldsymbol{x}^{[0]}, \ldots, \boldsymbol{x}^{[T]}$ where the final distribution $\boldsymbol{x}^{[T]}$ approximately is a Gaussian distribution with identity covariance (similar results hold for the binomial distribution).

Now the reversal of the diffusion process can be defined, i.e. the generative distribution with $\boldsymbol{x}^{[t-1]} \sim p(\boldsymbol{x}^{[t-1]}|\boldsymbol{x}^{[t]})$. It has been shown by Feller [47] that for small step size $\beta$ the conditional distribution $p(\boldsymbol{x}^{[t-1]}|\boldsymbol{x}^{[t]})$ will approximately be a Gaussian distribution. Hence, the chain $\boldsymbol{x}^{[T]}, \ldots, \boldsymbol{x}^{[0]}$ can be generated by a Gaussian distribution

$$\boldsymbol{x}^{[t-1]} \sim N(\boldsymbol{\mu}_{\boldsymbol{w}}(\boldsymbol{x}^{[t]}); \boldsymbol{S}_{\boldsymbol{w}}(\boldsymbol{x}^{[t]})) \quad \text{and} \quad \boldsymbol{x}^{[T]} \sim N(\mathbf{0}; \boldsymbol{I})). \tag{7.3}$$

This Gaussian distribution is completely defined by the mean and covariance of $\boldsymbol{x}^{[t]}$.

For the training, noisy samples $\boldsymbol{x}^{[t]}$ are generated by $q(\boldsymbol{x}^{[t]}|\boldsymbol{x}^{[t-1]})$ starting with the observed $\boldsymbol{x}^{[0]}$. From this the inverse $p(\boldsymbol{x}^{[t-1]}|\boldsymbol{x}^{[t]})$ may be reconstructed by optimizing the *variational lower bound* on negative log likelihood [65]. With the trained model one can start with a sample $\boldsymbol{x}^{[T]} \sim N(\mathbf{0}, \boldsymbol{I})$ and gradually reduce noise in a sequence of steps $\boldsymbol{x}^{[T-1]}, \ldots, \boldsymbol{x}^{[0]}$, where

$$\boldsymbol{x}^{[t-1]} \sim p(\boldsymbol{x}^{[t-1]}|\boldsymbol{x}^{[t]}) \approx N(\boldsymbol{\mu}_{\boldsymbol{w}}(\boldsymbol{x}^{[t]}); \boldsymbol{S}_{\boldsymbol{w}}(\boldsymbol{x}^{[t]})). \tag{7.4}$$

The distributions $p(\boldsymbol{x}^{[t-1]}|\boldsymbol{x}^{[t]})$ may be estimated conditional to image classes [37]. Instead of a finite number of image classes one may even use a caption text as condition. The text is first encoded into a sequence of $k$ tokens and fed into a Transformer model. The Transformer outputs a class embedding as well as $k$ token embeddings, which are used as additional model inputs. Here a normal noise term $\epsilon_{\boldsymbol{w}}(\boldsymbol{x}^{[t]}|\emptyset)$ for reconstruction is estimated and in addition conditional to the caption $c$ a noise term $\epsilon_{\boldsymbol{w}}(\boldsymbol{x}^{[t]}|c)$. During the *classifier-free reconstruction* both terms are mixed.

The diffusion model is approximated by a *U-Net* model [144] with 2.3B parameters, performing a downsampling of the 64 pixel image to a smaller resolution with many features and a subsequent upsampling. An additional 1.5B parameter model is used for upsampling to a $256 \times 256$ resolution. The caption text is processed by a transformer model with 1.2B parameters and the final token embedding is used in place of a class embedding.

In tests, GLIDE produced high-quality images with realistic reflections, textures, and shadows. The model can also combine multiple concepts (for example, dragon, psychedelic, and hamster) and attach attributes like colors to these concepts. On the MS-COCO benchmark with $256 \times 256$ images DALL-E achieves a FID-value of 28, while LAFITE gets 26.9 and GLIDE 12.2. Also in human evaluations, the results of GLIDE are clearly preferred. This is remarkable as GLIDE has far less parameters than DALL-E. Figure 7.14 shows some images generated by GLIDE. GLIDE can also be used for restoring a masked image patch according to a textual prompt, e.g. *"tie with black and yellow stripes"*. In most cases, GLIDE produces better results than competitor models and the corresponding image patch is restored with realistic


**Fig. 7.14** Images generated by GLIDE [109] according to the captions in the lower row. The best of a sample of 60 is shown. Image reprinted with kind permission of the authors [109, p. 7]

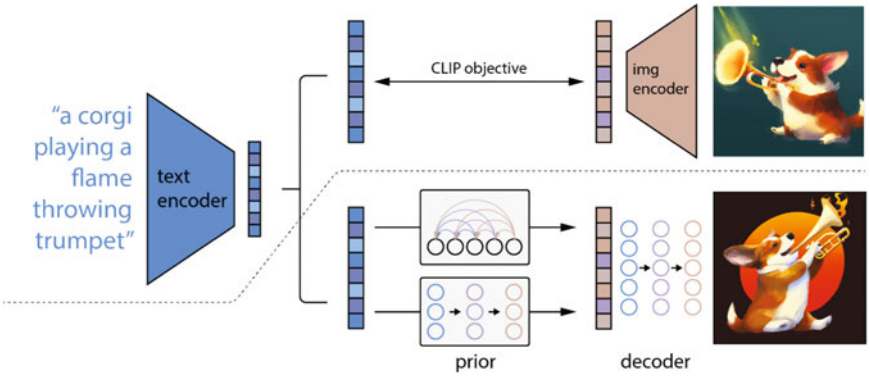


**Fig. 7.15** A high-level overview of DALL-E 2 [132]. Above the dotted line the CLIP training process is shown minimizing the difference between the embeddings for an image and the corresponding text. Below the dotted line, the text-to-image generation process is illustrated: a CLIP text embedding is first fed to an autoregressive transformer (higher box) or diffusion prior (lower box) to produce an image embedding. This embedding is used as input to the diffusion decoder which produces a final image. Image reprinted with kind permission of the authors [132, p. 3]

lighting, shadows and textures. Finally, GLIDE can add shadows and reflections to images and transform simple line sketches into photorealistic images.

**DALL-E 2** [132] is an improved version of DALL-E that can create more realistic art and images from a descriptive sentence in natural language. It works in two steps (Fig. 7.15): first a CLIP (Sect. 7.2.4) image embedding $z_i$ based on a text description $y$ is generated according to a prior $p(z_i|y)$. Then a diffusion-based decoder generates an image $x$ conditioned on an image embedding $z_i$. The decoder $p(x|z_i, y)$ inverts the CLIP image encoder, is non-deterministic, and can produce multiple images corresponding to a given image embedding. The CLIP model is frozen during training of the prior and decoder. The dimensionality of the image embeddings $z_i$ is reduced to 319 from 1024 by principal component analysis while preserving nearly all information. Each of the 319 dimensions is quantized

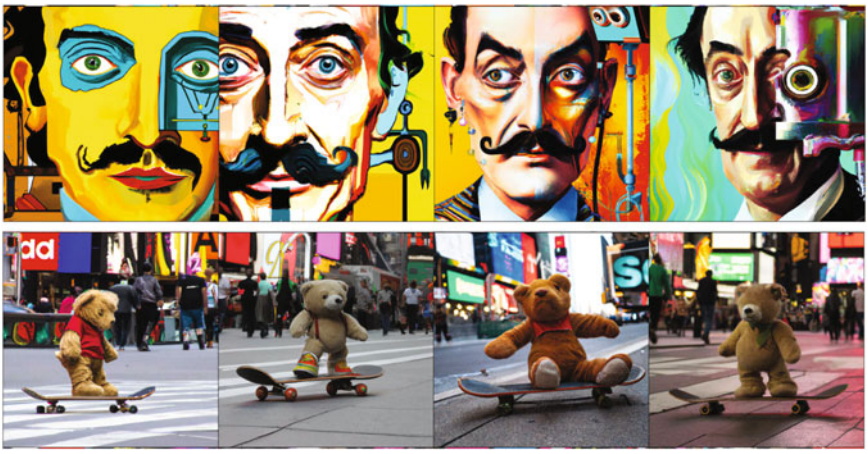

**Fig. 7.16** Random samples from DALL-E 2 [132] for the prompt *"Vibrant portrait painting of Salvador Dali with a robotic half face"* (upper row), and *"A teddybear on a skateboard in Times Square"*. Image reprinted with kind permission of the authors [132, p. 25,27]

into 1024 discrete buckets. For the encoder, experiments are performed with both autoregressive and diffusion models for the prior. It turns out that diffusion models are computationally more efficient and produce higher-quality samples. Examples are shown in Fig. 7.16.

The decoder is conditioned on image representations and can produce variations of an image that preserve both its semantics and style, while varying the nonessential details that are missing from the image embeddings. CLIP's shared embedding space allows for language-guided image manipulations and modifications in a zero-shot manner. For example two images $x_1$ and $x_2$ can be blended, interpolating all of the concepts in CLIP's embedding space that occur between them. With respect to MSCOCO it turns out that DALL-E 2 has a better zero-shot FID of 10.4 than GLIDE (12.2). Human comparisons show that DALL-E 2 and GLIDE are similar in terms of photorealism and caption similarity, while DALL-E 2 produces images with greater diversity. DALL-E 2 struggles more than GLIDE with a prompt that requires it to connect two separate objects (cubes) to two separate attributes (colors). A public access to DALL-E is now available for users to create images [115].

**Imagen** [148] is a text-to-image model presented by Google. It encodes the input text into text embeddings by a pre-trained T5-XXL encoder-decoder Transformer with 4.6B frozen parameters. A conditional text-to-image diffusion model (7.3) maps the text embeddings into a $64 \times 64$ image. Subsequently these small images are upsampled in two steps to $256 \times 256$ and to $1024 \times 1024$ by two super-resolution diffusion models with 600M and 400M parameters (Fig. 7.17). The models are trained on 860M image-text pairs.

Nichol et al. [110] proposed some modifications for denoising diffusion probabilistic models, which can sample much faster and achieve better log-likelihoods

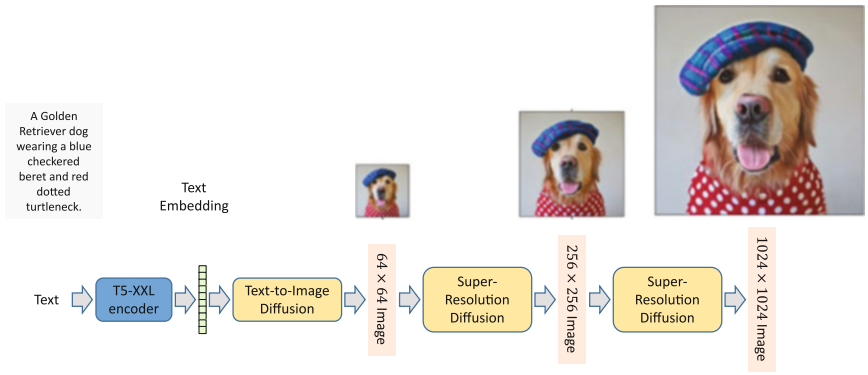


**Fig. 7.17** Imagen encodes the input text by the pre-trained T5-XXL text encoder. The resulting text embeddings are transformed to 64 × 64 images by a diffusion model [148]. This image is upscaled to 1024 × 1024 resolution by two super-resolution diffusion models. Image reprinted with kind permission of the authors [148, p. 19]

with little impact on sample quality. They deliver the same sample quality as GANs, but achieve a much better mode coverage as measured by recall. This model is also employed by Imagen for text-to-image conversion, using the pooled embedding vector as input. This network is used for upsampling and is extended to improve memory efficiency, inference time, and convergence speed. Figure 7.18 shows randomly selected images generated by Imagen for a caption input.

Imagen achieves a SOTA zero-shot FID (Fréchet Inception Distance) on *COCO* with a value of 7.3, which is better than the FID of DALL-E 2 and is even better than other models trained on COCO (Table 7.2). Human raters evaluated Imagen with respect to photorealism and alignment to the text caption. For photorealism, people preferred Imagen images in 39.5% of cases to the original images, indicating a relatively high realism. On caption similarity, Imagen's score is on-par with the original reference images. On the *DrawBench* [147] the images generated by Imagen are always preferred to images created by DALL-E 2, GLIDE, VQGAN+CLIP or Latent Diffusion in more than 60% of the cases. The authors emphasize that in the future they will increase the size of the language model, as this promises a greater gain than increasing the size of the diffusion models. They do not publish Imagen's code or provide a demo API because it could potentially be abused, for example to create fake images. Gafni et al. [48] demonstrate how a system can be extended to support artists during the creation of images.

**Stable Diffusion** is another model with currently 5.7B parameters for generating images of up to 1024 × 1024 pixels using diffusion. An example is shown in Fig. 7.18. It works similar to DALLE-2 employing a denoising U-Net for image compression and expansion [142]. For training, Stable Diffusion used an image dataset from the freely available LAION-5B database [12], which contains about 5.85 billion CLIP-filtered image-text pairs, fourteen times larger than its predecessor LAION-400M. A model conditioned on ImageNet classes achieved


**Fig. 7.18** Images generated by Imagen [148, p.6] (left) and Stable Diffusion [142] (right) given two different text captions. Images reprinted with kind permission of the authors [148, p. 6] and [158], credits in Table A.3

an FID of 3.6 for image generation. A variant of the model employs an image search returning images with similar visual features from the neighborhood of each training instance by the CLIP model [15]. The model includes the retrieved images during image generation. It can be applied to unconditional image synthesis, inpainting, and stochastic super-resolution, and achieves competitive performance while significantly lowering computational cost. Model inference code and model weights to run the retrieval-augmented diffusion models are now available [141] and can be downloaded. The model was heavily employed by users creating 1.7M images per day.

### *7.2.8 Multipurpose Models*

**OFA** (One For All) [170] provides a unified model for a range of multimodal tasks. It can process text and images in the form of text and visual tokens. OFA has an encoder-decoder transformer architecture (Sect. 2.3.1) and is pre-trained on various text and image datasets. Similar to the T5 model (Sect. 3.1.3), it receives a textual instruction along with an image and generates the appropriate output.

Different modalities are represented in the same space, and text, images, and objects are discretized into a unified output vocabulary. An image with $256 \times 256$ pixels is represented as $16 \times 16$ image patches. Each image patch of $16 \times 16$ pixels is "tokenized" into discrete visual tokens, such that each visual token strongly correlates to the corresponding patch [11]. In addition, objects have a specific representation consisting of a label and its bounding box. The continuous corner coordinates of the bounding box are uniformly discretized to integers as location tokens $(x_1; y_1; x_2; y_2)$. Finally, a unified vocabulary is used for all linguistic and visual tokens, including subwords, image codes, and location tokens.

Similar to T5 (Sect. 3.1.3) the transformer encoder-decoder is controlled by instructions. It receives a text instruction and an input image and generates a corresponding output, a text response and an image. A number of tasks are described by the examples shown in Fig. 7.19. Usually, the OFA model is fine-tuned on specific datasets to solve various tasks.

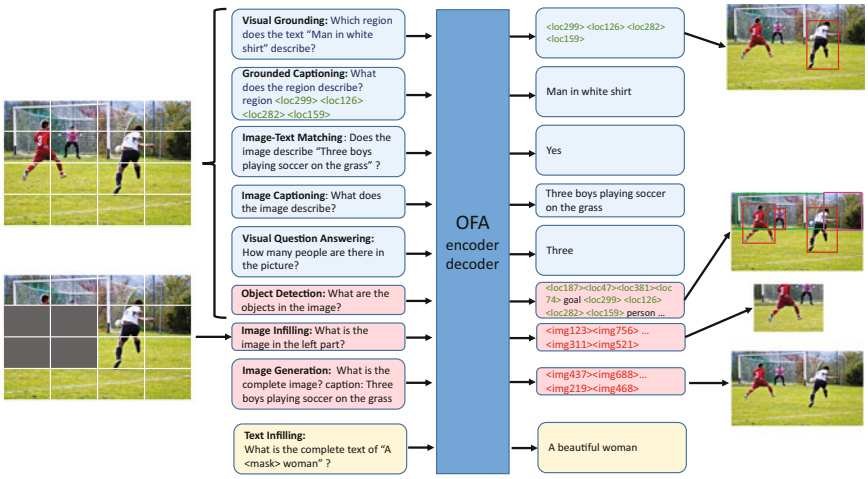


**Fig. 7.19** OFA [170, p. 3] receives an instruction and an input image. As output it generates a text and (optionally) an image. For each of the eight instructions (left) an example output (right) is shown. Image credits in Table A.3

The OFA model has an $\text{OFA}_{\text{Base}}$ variant with 6 encoder and decoder layers, hidden size 768, and 12 attention heads. The $\text{OFA}_{\text{Large}}$ variant has 12 encoder and decoder layers, hidden size 1024, 16 attention heads and 472M parameters.

During pre-training, the model has to solve three tasks requested by the corresponding instructions (Fig. 7.19). The first task is image infilling, where the model has to reconstruct the central parts of the image. This requires the model to learn the relation of image parts and the generation of images. The second task is object detection. This task establishes the correspondence between image parts and language descriptions. The last pre-training task is text infilling to learn the structure of language. The model is pre-trained on publicly available datasets for the different tasks on data with more than 50M images and more than 160GB text. Images are resized to $384 \times 384$ pixels with a fixed patch size of $16 \times 16$ pixel. For each patch a feature vector is computed by the first three blocks of a ResNet CNN.

Fine-tuning is performed on task-specific datasets for the tasks shown in Fig. 7.19, e.g. MS COCO for image captioning. In addition, OFA is fine-tuned on several NLP tasks such as the GLUE benchmark for natural language understanding, the Gigaword benchmark for abstractive summarization, and the ImageNet-1K dataset for image classification. For inference the authors apply beam search and develop a search strategy based on a prefix tree. This trie-based search strategy ensures that the output generated by OFA is constrained to the appropriate candidate set.

For image captioning the model is fine-tuned on MS COCO [26]. With a BLEU-4 score of 43.5 it establishes a new SOTA for the MS COCO benchmark [32]. For Visual Question Answering the model is fine-tuned on VQAv2 [56] and similar

datasets. A search strategy based on a prefix tree ensures that the output generated by OFA is constrained to the candidate set. It achieves a new SOTA accuracy of 80.0%.

For the *visual entailment task* the model has to determine, if the image entails, contradicts or is neutral to the text. OFA is fine-tuned on *SNLI-VE* [178] and achieves a SOTA accuracy of 90.2% on the test set, which is 3.1% better than the prior best model. To understand referring expressions, the model has to locate an image region described by a language query. Here the model was fine-tuned on the *RefCOCO benchmark* [187] and related benchmarks. It achieved a new SOTA with a text accuracy of 92.9%, outperforming competitors by a large margin.

For image generation the model is fine-tuned on MS COCO [26]. It achieves an Fréchet Inception Distance (FID) of 10.5. This is better than the scores for DALL-E [133] (27.5) or GLIDE [109] (12.2), which have far more parameters (12B resp. 3.5B) than OFA with 472M. On the leaderboard, only LAFITE (Sect. 7.2.6) has a better FID-value of 8.1. Note that competing models selected their results from 60 to 512 trial outputs, while OFA only selected the best of 24 images according to FID scores.

For image classification in ImageNet, OFA uses no extra labeled training data and has a similar performance (84.9% top-1 accuracy) as EfficientNet-B7 (84.3%), whereas the current SOTA is 88.3%. Surprisingly, OFA also achieves good results on language-only benchmarks, such as the GLUE natural language understanding benchmark (Sect. 4.1.1) and the Gigaword summarization (Sect. 6.4.1). Code, demos, and trained models are available for download.

An alternative multipurpose model is **NÜWA**, which is described in Sect. 7.3.4. It provides realistic text-to-image generation, image editing, and image region editing controlled by text. In addition, NÜWA performs text-to-video creation and the prediction of the next video frames.

**WuDao-2.0** [140, 143, 198] is a giant mixture-of-experts model with 1075B parameters and has been introduced in Sect. 3.5.2. It is based on the GLM 2.0 architecture (Sect. 3.1.3) combining the different learning paradigms of BERT, GPT and the encoder-decoder transformer. For image modeling, it uses the CogView approach (Sect. 7.2.6). However, implementation details are not available. The training data consist of 2.5TB image data and 2.5TB Chinese and English text data (e.g. from the *Pile* corpus [49]). WuDao-2.0 can be applied to a wide range of text analysis and generation tasks, and has matched or surpassed SOTA levels on five image benchmarks, e.g. on classifying land use in image data, image generation, and graphic retrieval.

### Available Implementations

- Vision transformer code, trained models and notebooks github.com/google-research/vision_transformer
- OSCAR code and pre-trained models github.com/microsoft/Oscar,
- VinVL code and pre-trained Oscar-VinVL models github.com/pzzhang/VinVL.

- DALL-E code and notebook github.com/openai/DALL-E
- OFA model code, pre-trained models and online demos github.com/OFA-Sys/OFA
- GLIDE code, trained models and notebook github.com/openai/glide-text2im
- Stable Diffusion https://github.com/CompVis/latent-diffusion

### *7.2.9 Summary*

Recently, the Vision Transformer (ViT) emerged as a competitive alternative to Convolutional Neural Networks (CNNs) for image recognition tasks. ViT models outperform CNNs in terms of accuracy on various benchmarks and require much less computational effort.

Foundation Models for image processing receive image patches as input. The embeddings of these image patches are generated by different methods, e.g. linear transformations of image pixels, by the first few layers of CNN models, by variational autoencoders (VAE), or by Generative Adversarial Networks (GANs). A completely different approach is taken by diffusion models, which reverse the process of image degradation by adding noise (GLIDE). It has been shown to be beneficial to discretize representations of image patches to reduce noise and low-level texture dependence.

There are two alternatives for including text. Sometimes text and image tokens are processed by separate transformers. Subsequently the distances between the two types of embeddings are minimized (CLIP) or the resulting embeddings are correlated by cross-attention (VilBERT). Otherwise, text and image tokens are concatenated to form the input of Foundation Models (autoencoders, autoregressive, or encoder-decoder). It seems that recent models (DALL-E, CogView, OFA) prefer the single-stream architecture. A number of different tasks are employed for pre-training. These include the masked language model (MLM), where masked image and language tokens have to be reconstructed, masked region classification (MRC), and masked region reconstruction. Sentence-image alignment (SIA) classifies whether image-text pairs belong together.

The generation of captions constructs a sentence with the characterization of the image (VilBERT, OSCAR, VinVL, SimVLM) in fluent and correct language, which is usually an accurate description according to human evaluations. The generation of longer captions is not yet investigated and is probably more relevant for video captioning. There are studies to investigate the attention patterns in vision-language models [19].

The creation of images that match captions has made a huge leap in quality over the past year. Various architectures are used: Generative Adversarial Networks (GAN), diffusion models, VAEs. These models are in general combined with PLMs. It seems that pure transformer models have advantages (OFA), but diffusion models like DALL-E 2.0 gain momentum. Usually, a sample of images is created, and the best image is automatically selected by a quality score. Images generated by the

model often have the resolution of $256 \times 256$ and already cover many details. Expect to see models with higher resolutions next year, e.g. $1024 \times 1024$.

Cao et al. [19] investigate the inner mechanics of vision and language models. They conclude that deeper layers lead to more intertwined multimodal fusion. Usually, the textual modality is more dominant for taking decisions than image features, as models tend to attend to text rather than images during inference. It turns out that a subset of attention heads is specialized for cross-modal interaction. There are attention patterns that align image regions and textual words. Finally, there is no reduction in linguistic capabilities, as pre-trained vision and language models encode rich linguistic knowledge.

Recently, multipurpose models have been presented that are trained to solve a large number of different language, vision, and language-vision tasks. One example is OFA, which has 472M parameters, significantly fewer than DALL-E (12B). OFA is a transformer encoder-decoder with image and text tokens as input, controlled by text instructions similar to T5. It achieves SOTA in image captioning, image generation, visual question answering, visual entailment, and even on pure language tasks. Contrast this with the huge WuDao 2.0 model with 1750B parameters, which is based on the encoder-decoder GLM model with a mixture-of-experts architecture. The model claims SOTA performance on a number of image and text tasks, but no technical details are known.

In the future, it is expected that these text-image models will be extended to other modalities such as video, speech, and 3D. In addition, more data will be used, Moreover, they will be enhanced by retrieval techniques to include additional external and up-to-date knowledge. Text-image models are a big step towards *symbol grounding*, which allows to attach symbols (words) to their real-world meaning.

## 7.3 Video Interpretation and Generation

As the Web is becoming a constantly growing communication vehicle, expressing content by text and images is often not sufficient. Video brings together three things that catch our attention like nothing else: image, movement, and audio. Therefore, videos are more and more important as a means of communication. There are 2 billion users active on YouTube each month and over 1 billion on TikTok with an average usage of 52 min per day. Hence, the automatic analysis, interpretation, and generation of videos is extremely valuable. For visual data, the most comprehensive self-supervision is available in videos. Their various modalities such as images, speech, ASR text, and captions are temporally aligned and do not require human annotation. The extreme number of multimodal videos potentially allows Foundation Models to acquire a model of the visual world.

### 7.3.1 Basics of Video Processing

Video analysis and understanding is more challenging than image processing, because video has an additional time dimension and usually has to handle images, speech, and text from speech or subtitles simultaneously. Recently Foundation Models have been used for video understanding. Compared to CNNs and RNNs, the major advantage of transformers is the ability to simultaneously capture global information and compute this in parallel. Furthermore, the concise and stackable architecture of transformers enables training on larger datasets. Table 7.3 list the main variants of Foundation Models for video.

Early models for image processing, e.g. CNNs and GANs, performed the analysis of images pixel-by-pixel. However, this is no longer possible for videos due to the high computational and memory effort, and there has to be an aggregation of image information. Therefore, special spatio-temporal aggregation modules are developed to adapt this to the limited sequence length of transformers.

- A simple solution is the aggregation of 30 video frames (VideoBERT).
- Videos can be processed by considering 3D *video patches*, which cover a small pixel region in a small number of frames. It is possible to aggregate video and text over different temporal levels and compute associations between the levels (COOT, MTV). Regional and temporal aggregation may be separated (CoVeR).
- Additionally the video patches may be processed to extract salient information. An example is video quantization by variational autoencoders (VQ-VAE), which already was used for image processing, e.g. by DALL-E or CogView (Sect. 7.2.6). Image patches can be extended in time to obtain 3D voxels (VATT, Omnivore).
- A video can be partitioned into short time clips. Prior clips can enter the self-attention computations but no update of prior embeddings is necessary (MeMViT).

To further reduce computational effort, a sparse self-attention can be used, where attention is mostly computed to nearby video pixels.

Unsupervised training may be performed similar to BERT. For instance, masked video tokens can be predicted based on neighboring video and text tokens [145]. In the same way, masked text tokens can be predicted from neighboring text and video tokens. Contrastive learning can be used to discriminate between genuine text-video pairs and random pairs. Other tasks include classifying whether a video and some text belong together, predicting the next frame, or reconstructing the order of shuffled video or text tokens. Recent surveys on video understanding are provided by Islam et al. [73], Khurana et al. [85], and Ruan et al. [145]

There are a number of training data sources for video. *Kinetics* [83] is a collection of 306k large-scale, high-quality datasets of 10s video clips focusing on human actions. The variants Kinetics 400, 600, and 700 are annotated with 400, 600, and 700 classes, respectively. Example frames of annotated videos are shown in Fig. 7.21. *Moments in Time* [107] is a collection of 800k labeled 3s videos, involving

**Table 7.3** Main techniques using PLMs for video. The numbers in parenthesis indicate parameter count

| Model | Approach | Benchmark |
|---|---|---|
| **Video to text** | | |
| VideoBERT | Partition video into 30 clips and generate embeddings by CNN. Cluster embedding by $k$-means. ASR speech generates text tokens. Concatenate inputs to BERT | YouCook II video captioning 4.3 BLEU-4 |
| COOT | Image, video and text are processed in 3 different hierarchy levels. Separate transformers for each level. Special attention for cooperation in each level (10.6M) | YouCook II video captioning 11.3 BLEU-4 |
| DeCEMBERT | Video 2D and 3D features, region captions, ASR text. Inputs linearly transformed and fed into a single BERT | YouCook II video captioning 11.9 BLEU-4 |
| VATT | Generate image-time patches, separate BERT models for video, audio, and text. Contrastive estimation to reduce embedding distances | Kinetics-400 action recognition 81.1% |
| Omnivore | Image, video and 3D views are converted and fed into Swin transformer with shifted windows | Kinetics-400 action recognition 84.1% (no extra data) |
| MeMViT | Attention computation with memory of past video frames. Memory not trained. Uses memory compression module with pooling | Action recognition on EPIC-KITCHENS-100 accuracy 48.4% |
| CoVeR | Separate image and temporal aggregation. Parallel fine-tuning for image and video recognition | Kinetics-400 action recognition 87.2% |
| MTV | Temporal aggregation by multiple views. Use different Vision Transformers for each view (1B) | Kinetics-400 action recognition 89.1% |
| Merlot | Joint processing of video and ASR text. MLM for text and video. Reorder scrambled frames | Visual question answering 43.1% |
| Flamingo | Process images, video by vision transformer (80B). Include image information into language model (Chinchilla) by adapters and cross-attention layers. Allows few-shot prompts | SOTA on all of 8 image benchmarks and all of 8 video benchmarks |
| **Text to video** | | |
| Video transformer | Partition video to 3D blocks with varying dimensions in different layers (373M) | AR video generation FVD score 94 on BAIR Robot data |
| NÜWA | Image, video and text data are represented as 3D tokens. Discretized by VQ-GAN. Use localized attention computations. Trained for text-to image, video prediction and text-to-video. More applications | AR video generation FVD score 86.9 on BAIR Robot data (SOTA) text-to-video FID-img 28.5 on Kinetics |
| Imagen video | Base video generation model + several spatial and temporal video super-resolution diffusion models | FVD score of about 9.0 for the model with 5.6B parameters |

people, animals, objects or natural phenomena that capture the gist of a dynamic scene. *Epic-Kitchens-100* [33] consists of 90k egocentric videos, totaling 100 h, recorded in kitchens. Each video is labeled with a "noun" and a "verb". Three accuracy scores ("noun", "verb", and "action") are usually reported. The action score assesses correct noun-verb pairs and is most important. *Something-Something V2* [55] consists of more than 220k short video clips that show humans interacting with everyday objects. Similar objects and backgrounds appear in videos across different classes. This data challenges a model's capability to distinguish classes from motion cues, in contrast to other datasets.

### 7.3.2 Video Captioning

*Video captioning* aims at automatically generating natural language descriptions of videos. Video captioning is substantially more difficult than image captioning because the spatial-temporal information in videos as well as the corresponding ASR text from the video introduces an additional complexity. On the other hand, huge video collections like YouTube are available on the Internet and can be used as training material. A recent survey is given by Perez-Martin et al. [124].

**VideoBERT** [160] applies a BERT model to video-text pairs. The video is partitioned into clips of 30 frames (1.5sec) and processed by the S3D CNN with a temporal convolution [180], which generates a clip embedding vector of size 1024. The clip embeddings are partitioned by $k$-means clustering into 20,736 clusters and quantized to video tokens. Speech is processed by ASR and partitioned into sentences. The text is tokenized by WordPiece with a vocabulary of 30k tokens. The video tokens corresponding to the sentence time period are collected in a video token sequence. As shown in Fig. 7.20 the video tokens are appended to the corresponding text tokens separated by special tokens. Note that text-only and video-only training is possible as well.

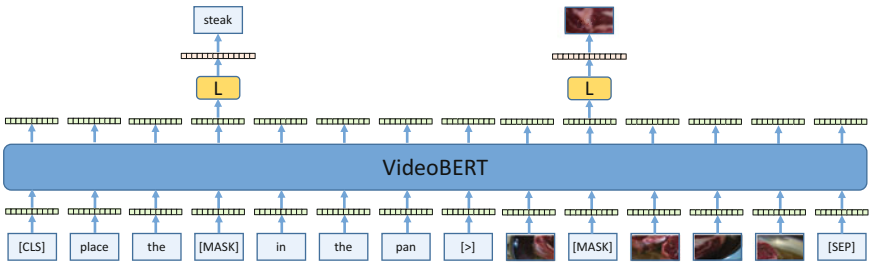


**Fig. 7.20** A text generated by ASR and the corresponding video tokens are the input of VideoBERT [160]. Both modalities are bounded by special tokens. The masked tokens have to be predicted. Image credits in Table A.3

The $\text{BERT}_{\text{LARGE}}$ model is pre-trained on a video set of 312k cooking videos with a total duration of 966 days. The text is obtained by ASR. Training tasks are masked token and frame prediction, and detecting text matching a video. VideoBERT yields SOTA on video captioning on the YouCook II data with BLEU-4 score of 4.3.

**COOT** [51] jointly processes image, video and text information with an universal representation by embedding vectors. In the representation of videos, time is added as a third dimension to the two-dimensional description of images. The COOT model considers the data on 3 different levels of hierarchy: frame/word, clip/sentence and video/paragraph. For each level there exists a pair of transformers processing the input. To model intra-level cooperation, COOT uses a feature aggregation layer to focus on temporal interactions between low-level entities. To aggregate information to the sentence level, the model uses a special attention formula, where all corresponding embeddings enter the scalar product. An additional loss term aims to reduce the difference between sentence and clip encodings. At the top level, a contextual transformer links the text and video embeddings.

The model is trained with videos that have subtitles for individual scenes and longer segments. Subsequently, the model can create subtitles for new videos. For the YouCook2 video subtitling benchmark dataset, the model can greatly improve the SOTA to 11.3 BLEU-4. In addition, the model can also be used for other tasks, such as searching when a textual description or a video scene is entered. Since the model includes only 10.6M parameters, it is expected that performance can be greatly improved by increasing the size of the model.

**DeCEMBERT** [164] aims to enhance a video by *region captions* in addition to the ASR-text extracted by speech recognition. The input text is represented by BPE-tokens. Each second of video is characterized by 2D-features extracted by a pre-trained Resnet-152 CNN [63] as well as by motion features extracted by a 3D ResNeXT CNN [179], which together are mapped to embedding vectors. The video embeddings and speech recognition text representations are concatenated forming a single sequence as inputs to a 12-layer autoencoder for pre-training and downstream task fine-tuning. To align video with ASR captions, a constrained attention loss is used that encourages the model to select the best matched ASR caption from a pool of candidates. During pre-training on 1.2M YouTube instructional videos, the association between text and video is learned by masking tokens and by a classification, if a text corresponds to a video. On the YouCook2 captioning task the model improves SOTA to a BLEU-4 score of 11.9. In addition, DeCEMBERT yields good results for video retrieval and video question answering.

### *7.3.3 Action Recognition in Videos*

**VATT** [2] uses raw RGB frames of Internet videos, audio waveforms, and ASR text of the speech audio as input data. The video of size $T \times W \times H$ with $T$ frames is partitioned to a sequence of $\lceil T/t \rceil * \lceil H/h \rceil * \lceil W/w \rceil$ patches, where each patch is a *voxel* in $\mathbb{R}^{t \times h \times w \times 3}$ with an additional color dimension. This is an

extension of the image patches of ViT. The position encoding is a sum $\boldsymbol{e}_{i,j,k} = \boldsymbol{e}_{\text{temp};i} + \boldsymbol{e}_{\text{horiz};j} + \boldsymbol{e}_{\text{vert};k}$ where each of the summands is a learnable vector of length $d$. The raw audio waveform is partitioned into $t'$ segments and each segment gets a learnable position embedding. For the text a vocabulary is created and each word is mapped to a learnable embedding. The *DropToken* procedure removes a random sample of the video or audio tokens to reduce computational cost and improve regularization.

VATT linearly projects each modality into a feature vector of length $d$ and feeds it into a separate BERT encoder. The model uses Noise Contrastive Estimation to reduce the distance between projections of the audio and video embeddings. Positive pairs are taken from the same location in the video, and negative pairs from different locations. A similar criterion is employed to reduce the distance of video and text embeddings. The training data covers clips of 32 frames at 10 fps taken from the *HowTo100M data* [105]. The largest model has 415M parameters. For action recognition on Kinetics-400 it achieves SOTA with a top-1 accuracy of 82.1% and a top-5 accuracy of 95.6%.

**Omnivore** [52] is a model for classifying images, videos, and single-view 3D data using exactly the same model parameters. A single-view 3D is a color image with an additional depth channel. Image, video, and single-view 3D modalities are converted into embeddings that are fed into a Transformer model. The images are partitioned into image patches, videos are divided into spatio-temporal tubes covering separate image regions, and the single-view 3D images are converted into RGB patches and depth patches. The patches are projected into embeddings using linear layers. The same linear layer is used for image and video RGB patches. A separate layer is applied to depth patches. Separate positional embeddings for the spatial and the temporal dimension are used.

Omnivore employs the *Swin transformer* (Sect. 7.2.3) as base model, a hierarchical vision transformer using shifted windows. Self-attention involves patch embeddings from spatially and temporally nearby patches. The models are jointly trained on the ImageNet-1K dataset for image classification (1.2M images), the Kinetics-400 dataset for action recognition (240k videos), and the *SUN RGB-D dataset* (5k) for single-view 3D scene classification, with dataset-specific linear classification layers transforming the final embeddings. On Kinetics-400 without extra data, Omnivore achieved an action recognition accuracy of 84.1%, which was the second best. The fine-tuned Omnivore scored SOTA on two video classification benchmarks. When using RGB and the 3D channel, Omnivore again had a SOTA performance on the NYU-v2 benchmark.

**MeMViT** [173] aims to process videos longer than 5s, in contrast to most current models. MeMViT handles videos in an online fashion and caches key and value vectors of a transformer as memory at each iteration. Through the memory, the model has access to prior context for long-term modeling, with little additional cost, as memory embeddings are not trained. The queries of the current video clip attend to an extended set of key and value vectors, which come from both the current time and the past. Similar to the dilated convolutions of WaveNet [114], higher layers attend further down into the past, resulting in a significantly longer receptive field.


**Fig. 7.21** Two videos annotated with descriptions (left) similar to videos of the Kinetics dataset [83]. Representative frames of the videos are shown. Obviously, a single frame is sometimes not enough to reach a decision, e.g. to distinguish "dribbling basketball" and "dunking basketball". Image credits in Table A.3

In addition, a memory compression module with learnable pooling is effective for reducing the memory footprint.

A video is split into a sequence of short $T \times H \times W$ clips and processed sequentially. Similar to MTV, multiple resolutions are used, starting from a fine-grained modeling of smaller patches to high-level modeling of larger patches in later stages, where the dimensionality of embeddings increases. The aggregation between stages is done by strided pooling. The memory representations are frozen and not changed by optimization. The model is pre-trained on Kinetics-400 data Fig. 7.21. On the AVA v2.2 dataset [54] MeMViT achieves a mean average precision (mAP) of 35.4%. On the action anticipation dateset (EPIC-KITCHENS-100) it has a SOTA of 17.7% recall@5. On the action recognition on EPIC-KITCHENS-100 MeMViT yields an accuracy of 48.4%.

**CoVeR** [190] evaluates the effect of different pre-training strategies on classification accuracy. The authors use a special transformer architecture, which has spatial attention layers across related regions in the same video frame and temporal attention layers across the neighboring frames of a video clip. CoVeR first pre-trains the model on the *JFT-3B benchmark* [189] of 3B images annotated with a class-hierarchy of around 30k labels. During pre-training all temporal attention layers are removed. During fine-tuning, it simultaneously trains a single model with 24 layers on multiple action recognition and image datasets (Kinetics versions, ImageNet, Moments in Time, SomethingSomethingv2) to build robust spatial and temporal representations of video data (Fig. 7.22). For the Kinetics-400 action recognition task CoVeR achieves an accuracy of 87.2% and for the Moments in Time action classification it has a SOTA accuracy of 46.1%.

**MTV** [185] performs temporal aggregation by multiple input representations (views) of the input video. MTV extracts tokens from the input video over multiple time spans. Video tokens derived from long time intervals capture the overall scene description, while video tokens taken from short segments capture fine-grained

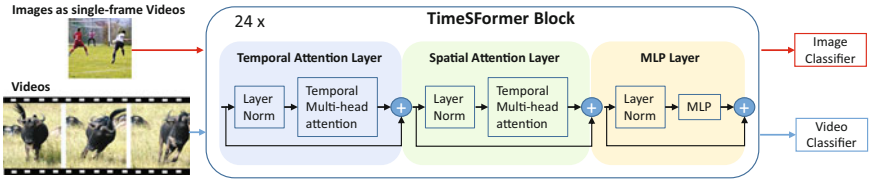


**Fig. 7.22** During fine-tuning CoVeR [190, p. 5] simultaneously is trained on multiple image and video datasets. Each dataset has its own classifier as there are different class definitions. Images are single frame videos. Therefore, image classification is not affected by temporal attention. Image credits in Table A.3

details, such as a person's gesture. Different transformer encoders are used to process these different views, with short segment models having higher capacity.

The different encoders are interfaced by lateral connections to fuse cross-view information. A cross-view attention is computed between adjacent views similar to the multi-head cross-attention in the transformer (Sect. 2.3.1). Note that these fusion operations are performed only for specific layers. The tokens from all views are aggregated with a global encoder, which performs the final classification.

The models are initialized with Vision Transformer weights (Sect. 7.2.2) and trained with videos of 32 frames and a resolution of $224 \times 224$. It turned out that the cross-view attention was better than alternatives to fuse information from different views. In addition, three views gave better results than fewer views. The largest model with over a billion parameters achieved SOTA accuracy of 89.1% for action recognition on kinetics-400.

**AV-ASR** [152] applies a PLM to audio-visual speech recognition. As usual, audio is converted to 80 log Mel filterbank features in steps of 10 ms. The videos are cropped to a near mouth region and converted to video embeddings with length 512. Both embeddings are concatenated and fed into a Conformer encoder (Sect. 7.1.2) with 17 layers. The model outperforms previous SOTA for lipreading on the LRS3-TED benchmark [1] with a WER of 19.3%. If both modalities are used, the WER drops to 1.6%. If babbling noise is added the WER of audio-only ASR on LRS3-TED is increased to 6.1%, while speech recognition with both modalities has a WER of only 2.9%. There is another approach to associate video and audio by generating video background music that matches the speed of movement, mood, and rhythm of the video [38].

**Aloe** [39] wants to do more than simply describing an image or video, but aims at explaining or reasoning about the scene. The model uses an unsupervised object segmentation module that partitions each image into object representations. A transformer receives the questions and the image descriptions including object representations. On several *visual reasoning* benchmarks, the model has to answer complex question such as explanatory questions like *"why did something happen?"*, predictive questions such as *"what will happen next?"*, and counterfactual questions like *"what would happen in an unseen circumstance, e.g. if an object is removed?"*. The model is able to improve SOTA on nearly all benchmark dataset.

**Merlot** [188] is a vision and language model that learns multimodal world representations from videos with thousands of frames and their ASR text. It encodes each frame using an image encoder, embeds tokens using learned embeddings, and a Transformer similar to RoBERTa jointly processes both representations. A first pre-training task uses contrastive loss to match the language transcript embedding and the corresponding video embedding. The MLM task requires replacing masked language tokens. The temporal reordering task involves reordering scrambled video frames. Hence, Merlot not only learns to match images to temporally corresponding words, but also to contextualize what is happening globally over time, achieving temporal common sense knowledge. The model is trained on 6M unlabeled YouTube videos. Merlot outperforms SOTA methods in 12 downstream benchmarks that include short and long videos. An example is Visual Question Answering on MSRVTT-QA [182] with a new SOTA of 43.1%. A related model for complex event extraction [93] uses a similar contrastive learning approach.

**Flamingo** [3] is a visual language model, which can handle sequences of arbitrarily interleaved image, video and text data. Flamingo employs the 70B parameter pre-trained language model *Chinchilla* trained on a large and diverse text corpus (Sect. 3.1.2). The encoder blocks of the language model are used with frozen parameters. With this submodel, Flamingo has strong generative language abilities and access to a large amount of knowledge stored in the Chinchilla weights. Similar to *Frozen* (Sect. 7.2.5), it can be instructed by few-shot learning to answer questions on an image [166].

For processing images and videos, a contrastive text-image approach is pre-trained (Fig. 7.23). The authors use a variant of ResNet [16]. The vision encoder is pre-trained using a contrastive objective on our datasets of image and text pairs, using the two-term contrastive loss from [127]. Much like CLIP (Sect. 7.2.4), similarities are computed as a dot-product of the mean pooled output of the image encoder and the mean pooled output of a BERT model. This model extracts semantic spatially oriented features from the image including color, shape, nature, positions of objects, etc. The model is pre-trained separately, and the parameters are frozen during the main training of Flamingo.

Two modules are trained to interface these frozen models. The first is a *perceiver resampler*, which receives spatio-temporal features from the vision encoder and outputs a fixed-size set of visual tokens (usually 64). This output is generated for single images as well as videos independently of the input image resolution or the number of input video frames. The extracted visual tokens are then included into the language model by interspersed cross-attention layers. In this way the language model can incorporate the visual information at each layer. The frozen language and vision models have 70B and 435M parameters, while the trainable layers have 10B parameters and the resampler has 194M parameters yielding a total of 80.6B parameters.

For training, Flamingo uses a number of datasets with 182GB of text. This collection is amended with further mixed text, image and video sequences with a total of about 2.3B images and 27M videos.

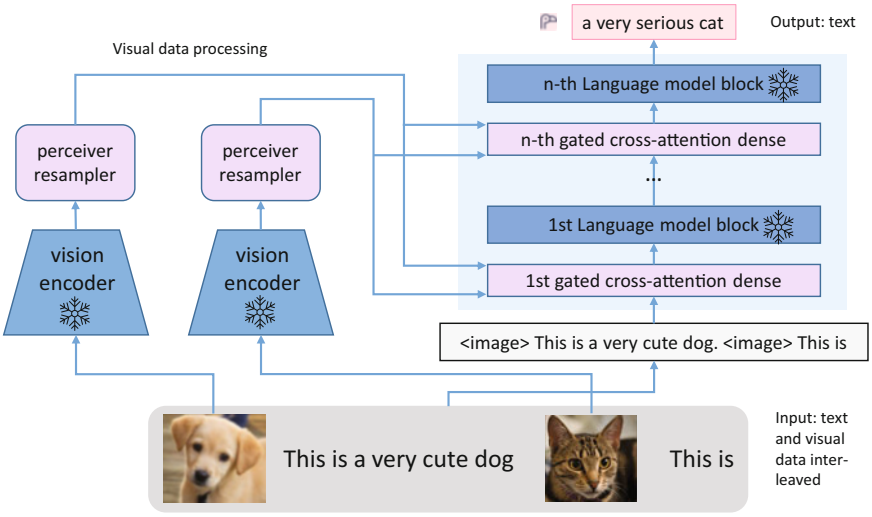


**Fig. 7.23** Flamingo [3] receives an input consisting of a sequence containing image, text, and video in arbitrary order (bottom). The images and videos are processed by a frozen vision encoder similar to CLIP. The trainable perceiver resampler reduces them to a finite number of image tokens, which are included by a trainable cross-attention layer into the language model. The output created by the language model is the natural continuation of the input sequence. Image adapted from [3] with kind permission of authors, credits in Table A.3

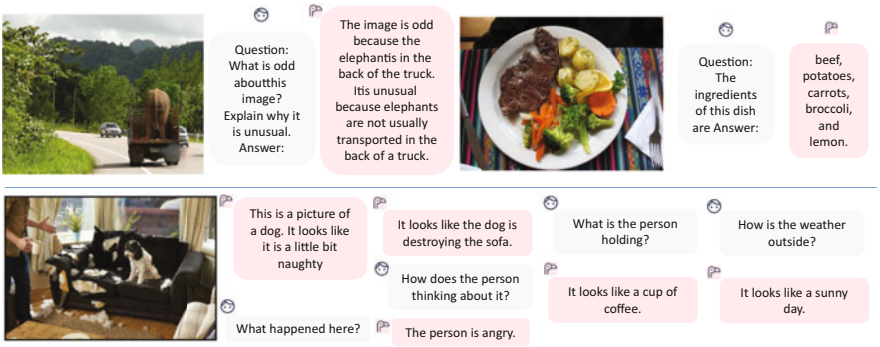


**Fig. 7.24** Flamingo can interpret images and describe them by text. Gray boxes are user input and the pink boxes are Flamingo output. In the upper row Flamingo answers questions about images. In the lower row there is a dialog about a photo. Image adapted from [3, p. 31] and [3, p. 32], reprinted with kind permission of the authors

As shown in Fig. 7.24 Flamingo can answer question on single images by simply predicting the next text token in the mixed image-text sequence. In their simplest form, the question can ask for the description of objects in the scene, as is shown in the upper right example. More difficult is the interpretation of the scene as the language model needs world knowledge to decide which aspects of an image are


**Fig. 7.25** Flamingo answers question on videos. Some video frames are shown. Gray boxes are user input and the pink boxes are Flamingo output. Image adapted from [3, p. 33], reprinted with kind permission of the authors

noteworthy. In many of these examples, Flamingo can do at least one step of implicit inference. Some of the objects are not named in the prompt (e.g. the elephant), but their properties are queried directly. In order to answer these questions, the model needs to infer the referred object and then recall the relevant knowledge to form the answer. This can lead to a single answer (as for the elephant on the truck) or to an extended dialog, where the model can answer a series of queries about an image (e.g. the dog damaging the sofa). Even after several interactions, Flamingo can still successfully attend to the image and reply to questions that require to interpret the image. The authors observed that multiple images can be separately attended to, simple comparisons and inferences are handled properly. Flamingo's dialog capabilities could enable non-expert end users to get answers without the need of fine-tuning.

In the same way Flamingo can answer question about videos, as shown in Fig. 7.25. However, the performance in this task is not as stable as would be desirable.

Flamingo is able to perform *few-shot prompting* on mixed text-video-image sequences. Examples are shown in Fig. 7.26. Here a number of images are provided and the added text specifies by example the desired way to extract an answer. In the first row this amounts to extracting text from the image and in the second row the counting of objects of equal type is required. In this way the model can be instructed on the fly to perform a large number of tasks, e.g. captioning, visual dialogue, classification or visual question answering.

The performance of the model was tested on 9 image-text benchmarks on scene description, visual dialogue, and visual QA, among them MS-COCO captioning. On the eight mixed-media benchmarks Flamingo established a few-shot SOTA by a wide margin using 16 or 32 shots. For three benchmarks the score is even better than the prior fine-tuned SOTA. On ImageNet top-1 classification Flamingo achieves 76.0% compared to a fine-tuned SOTA of 91.0%. The test array on video contains

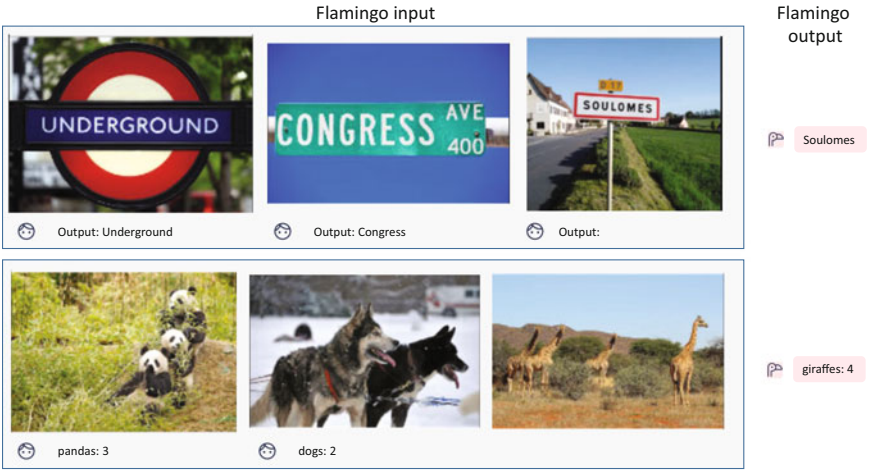


**Fig. 7.26** Few-shot querying of Flamingo [3] with a mixture of images and text. Note that in the second example Flamingo did not count the trees but stayed with the animals. The usual number of few-shot queries is 32. Image adapted from [3, p. 2], reprinted with kind permission of the authors

9 benchmarks, eight of whom require free form text answers and one benchmark (Kinetics 700) needs classification. On all eight free-form benchmarks Flamingo can increase few-shot SOTA, often by a huge margin. On four of these benchmarks Flamingo even exceeds the fine-tuned results. This is even more remarkable as Flamingo uses only 32 task-specific examples which is around 1000 times less task-specific training data than current state-of-the-art.

Flamingo can be fine-tuned on specific benchmarks to increase performance. During fine-tuning, the frozen model parts are not changed. When fine-tuning on 9 example tasks Flamingo could increase fine-tuned SOTA on five of these tasks. This shows that by fine-tuning the 10B free parameters of the model, the performance can in many cases be increase to new levels.

### *7.3.4 Generating Videos from Text*

The creation of videos following a textual description is an important issue, e.g. for education or illustration of dynamic content. While there are a number of models for describing images and videos through text, there are not many proposals for the other direction. The concepts for encoding text and videos are similar to the captioning of videos. The quality of generated videos can be judged by several measures comparing the similarity of actual and generated videos. The *FVD* (Fréchet Video Distance) is the spatiotemporal extension of the Fréchet Inception Distance (FID) (Sect. 7.2.6), and is sensitive to visual quality, temporal coherence and diversity of samples.

The **Video Transformer** [172] generalizes the one-dimensional transformer encoder-decoder to videos. A video is represented as $\boldsymbol{x} \in \mathbb{R}^{h \times w \times s \times d}$, where $h$ and $w$ denote the number of tokens in the spatial height and width, $s$ denotes the number of tokens in the temporal axis, and $d$ is the number of channels (e.g. colors). The video is partitioned into small 3D blocks in time and space. Self-attention is applied separately with each block. To allow direct information exchange between blocks, the block sizes between each layer are varied. The blocks contain 4 frames with a spatial resolution $32 \times 32$. Self-attention varies between 1 and 32 in different layers and dimensions. The largest model has a hidden size of 2048, 8 layers and 373M parameters. On the BAIR Robot Pushing data [44] the model achieved an *FVD* (Fréchet Video Distance) score [167] of 94. which was SOTA at the time of publication.

**NÜWA** [175] is a recent transformer encoder-decoder model that provides a solution for generating video from text. It uses a so called *3D Nearby Attention* mechanism to capture the locality characteristic for both spatial and temporal axes. Image, video and text data is represented as tokens $\boldsymbol{x} \in \mathbb{R}^{h \times w \times s \times d}$, where $h$ and $w$ denote the number of tokens in the spatial height and width, $s$ denotes the number of tokens in the temporal axis, and $d$ is the dimension of each token. The raw input regions are transformed into discrete tokens for image patches by a trainable VQ-GAN (Sect. 7.2.3). This GAN-based quantization module provides a much better image quality than VQ-VAE used by CogView (Sect. 7.2.6).

The model modifies attention computations and considers a local neighborhood with respect to width, height and temporal extent called 3D Nearby Self-Attention. Three different positional encoder embeddings are used for width, height and time. Each $336 \times 336$ pixel video frame is partitioned into $21 \times 21$ patches and 10 frames of a video are sampled with 2.5 frames per second. The size of the neighborhood in width, height and time is 3. The model is pre-trained on three tasks: Text-to-image for 2.9M text-image pairs from Conceptual Captions, video prediction with 727k videos from Moments in Time, and text-to-video generation for 241k text-video pairs.

For text-to-image generation, NÜWA is fine-tuned on the MS COCO dataset. Sixty images are generated for each text and the best image is selected by CLIP (Sect. 7.2.4). NÜWA outperforms CogView with an FID-0 of 12.9, which is good, as shown in Fig. 7.27, but worse than LAFITE (FID 8.1) and OFA (FID 10.5). For text-to-video, NÜWA is fine-tuned on the Kinetics dataset. Some frames of two generated examples are shown in Fig. 7.28. NÜWA achieves the best performance on the FID-img and FID-vid metrics with values of 28.5 and 7.0. Video prediction has to generate the sequence of the next frames of a video from a starting frame. On the BAIR Robot Pushing dataset, NÜWA achieves a new SOTA of 86.9 FVD score for this task.


**Fig. 7.27** 256 × 256 images generated from the text above the images by NÜWA [175] for the MS COCO benchmark. Image reprinted with kind permission of the authors [175, p. 5]


**Fig. 7.28** Frames of two videos generated by NÜWA [175] from text (left) for the text-to-video task on the Kinetics dataset. Note that an input text like "running on the sea" has never been seen by the model. Image reprinted with kind permission of the authors [175, p. 5]

NÜWA supports a number of other tasks. For image editing, it can reconstruct parts of an image. Alternatively, it can edit a marked image region according to a text, e.g. *"a horse is running on the grassland"*. Image sketches annotated with text are transformed to photos. This pattern can also be applied to videos, such that a video is generated from a series of images with annotated regions. Finally, it can change the contents in a video, e.g. modify the movements of a diver as shown in the lower row of Fig. 7.29. Moreover, a series of image sketches annotated with text can be transformed to a video. Further examples are shown here [174]. **GODIVA** [176] is a similar prior approach from the same authors based on VQ-VAE variational autoencoders.

**Imagen Video** is a recent high definition text-to-video model based on Imagen (Fig. 7.17). By a frozen T5 text encoder-decoder and a base video diffusion model a low-resolution video is generated. This is augmented by a cascade of video diffusion models that alternately increase spatial and temporal resolution [66] to construct 128 realistic video frames at 24 frames per second with a resolution of 1280 × 768. Figure 7.30 shows videos generated for text prompts by Imagen Video.

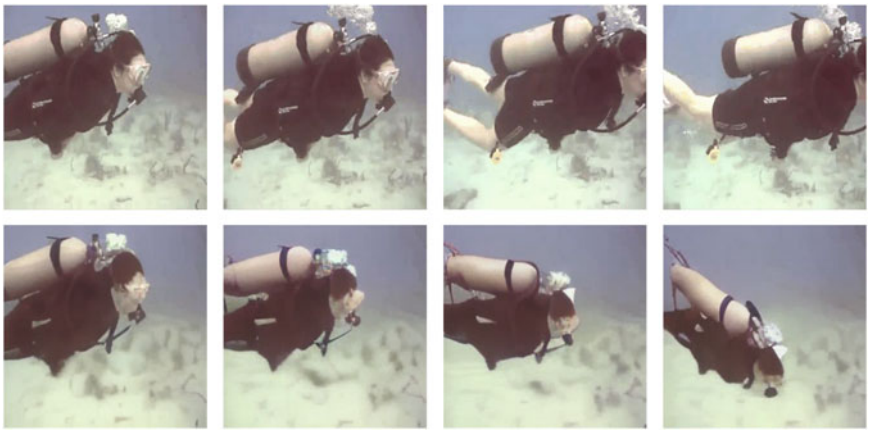

**Fig. 7.29** NÜWA [175] can edit videos. In the upper row the raw video is shown. In the lower row NÜWA gets the input "The diver is swimming to the bottom" and modifies the video accordingly. Image reprinted with kind permission of the authors [175, p. 28]


**Fig. 7.30** Videos generated from the text prompts (below) by Imagen video [66]. The model produces diverse and temporally coherent videos that are well matched to the given request. Image reprinted with kind permission of the authors [66, p. 2]

### Available Implementations

- VideoBERT code https://github.com/ammesatyajit/VideoBERT
- COOT code https://github.com/gingsi/coot-videotext
- DeCEMBERT code https://github.com/zinengtang/decembert
- VATT code https://github.com/google-research/google-research/tree/master/vatt
- Omnivore code https://github.com/facebookresearch/omnivore

- Video Transformer code https://github.com/rakhimovv/lvt
- MTV code and models https://github.com/google-research/scenic
- NÜWA code https://github.com/lucidrains/nuwa-pytorch

### *7.3.5 Summary*

The processing of videos requires to integrate different modalities like image, text in the form of video captions, and speech possibly translated to text by ASR. Video processing introduces an additional time dimension to image processing. Furthermore, depth information and camera movements can be important. Since 2019 large scale transformers using self-supervised pre-training are the prevailing models for video processing. The models can solve different tasks, such as video captioning, action recognition, video question answering, video generation from text, prediction of next frames, video retrieval, audio-visual ASR, etc.

Existing cross-modal Foundation Models mainly focus on (1) improving model architecture, (2) utilizing more data, and (3) designing better pre-training tasks. Due to the limited input length, the video has to be partitioned into appropriate tokens. This ranges from aggregates over 30 clips (VideoBERT) over fixed video patches (VATT) to video patches with varying dimensions (COOT, MTV, Video Transformer). Some models (VideoBERT, DeCEMBERT) use CNN convolutions to generate low-level features. More common is the aggregation with VQ-VAE autoencoders or the GAN-bases VQ-GAN. Sometimes video and text are processed with separate PLMs and merged later (VATT). Alternatively, video and text tokens are concatenated and processed by single a PLM (Omnivore, Merlot). Transformers use attention over spatial and temporal dimensions, which is often localized to reduce computational effort.

The integration of different modalities is crucial. Text and language are associated by pre-training tasks, where masked video or text tokens have to be predicted using tokens from the other modality. CoVeR shows that performance can be enhanced when the model is simultaneously fine-tuned for video and image tasks. It is even possible to combine audio, text and video tokens.

The performance of video analysis models has taken a dramatic development. The action classification error on the Kinetics-400 benchmark has fallen within 1 year to 10.9% using Foundation Models, which is a drop of 33%. Despite the significant progress, SOTA methods fail to extract/capture all the complex spatiotemporal information present in videos. There is still much work to do for understanding the diversity of visual content in videos and the structure of associated textual descriptions.

Generating videos from captions is in its early stages, and only very short high-resolution videos can be generated. However, the current models are relatively small compared to the Foundation Models like GPT-3 or Gopher. Therefore, it can be expected that models with more parameters will see considerable performance improvements, as has been demonstrated by Imagen Video.

There is a trend to general-purpose models, like Nüwa that can handle multiple modalities of data and solve a number of tasks. Training with different media mutually supports the performance in different tasks. Flamingo with 80B parameters is based on a large pre-trained language model and a separately pre-trained vision encoder. In can process mixed sequences of images, text and videos. By building adapter modules and a cross-attention layer, the language model can include the results of the visual modalities and perform a variety of analysis tasks like visual question answering, image caption, etc. In addition, it can be instructed by few-shot prompts to solve many task without a specific fine-tuning.

Although Flamingo cannot generate images or videos corresponding to a caption, it is a step in the direction of multimodal Foundation Models, which promise to be a general-purpose tool of multimedia processing. By few-shot prompts they could solve thousands or millions of tasks. Substantial progress can be expected in this area, as ideas can be combined that were developed independently for different media. Further development directions are larger training data, which, however, are already quite large. In addition, the development of multilingual video models is a logical consequence of the current state of research in this area.

## 7.4 Controlling Dynamic Systems

Foundation Models can process many types of sequences. These include sequential decision problems where the agent must choose an action based on a state. Subsequently, the environment generates a new state and a reward for the agent. This is repeated a number of times until the final sum of rewards is known. Then the task is to select the actions based on the states in such a way that the sum of rewards is maximal. This goal can be formulated as a sequence problem, and a PLM can be used to predict the next optimal action.

### *7.4.1 The Decision Transformer*

PLMs are able to predict sequences, e.g. the tokens of a text or video frames. Following this pattern, PLMs are also able to model the evolution of arbitrary states. *Reinforcement learning* considers a system with *states* $s_t$, *actions* $a_t$, and *rewards* $r_t = R(s_t, a_t)$ at a given time step $t$. Based on the current state, the agent selects an action, while the next state and reward are determined by the environment. The target of reinforcement learning is to learn a *policy* $a = \pi(s_t)$, which generates actions maximizing the expected sum of rewards $E(\sum_{t=1}^{T} r_t)$. During online reinforcement learning the environment can be accessed, and for a given $(s_t, r_t, a_t)$ it returns the next state $(s_{t+1}, r_{t+1})$. In offline reinforcement learning there is only a limited set of observed trajectories from the environment. The latter setting is more difficult as the agent can no longer explore the environment.

The **Decision Transformer** [23] operates in an offline reinforcement setting. Instead of using the returns $r_t$ directly, the Decision Transformer considers the *forward sum of rewards* $\hat{R}_t = \sum_{t'=t}^{T} r_{t'}$. Hence, a trajectory is represented as follows

$$\tau = \left(\hat{R}_1, s_1, a_1, \hat{R}_2, s_2, a_2, \ldots, \hat{R}_T, s_T, a_T\right) \tag{7.5}$$

The input token embeddings for $(s_t, r_t, a_t)$ are computed with a linear layer, which is different for each modality (Fig. 7.31). If the state is an image, it is transformed by a convolutional encoder instead of a linear layer. Subsequently the embeddings are normalized by a layer normalization. For each time step with three inputs a position embedding is learned and added to the embeddings of that time step. The embeddings are then processed by an autoregressive GPT model, which predicts future actions by autoregressive modeling.

The training was based on a dataset of observed trajectories. From these trajectories minibatches of length $K$ were sampled. Then the GPT model for each $t = 1, \ldots, K$ predicted $a_t$ given a trajectory up to $s_t$. As a loss function the cross-entropy loss was used for discrete actions with the target to increase the probability of the actual action at time $t$. For continuous actions, e.g. a speed, the mean squared error was used as loss to minimize the square difference to the observed control value. It was not necessary to predict states or the forward sum of rewards.

For the application to a starting state $s_1$, a target forward sum of rewards $\hat{R}_1$ based on the desired performance (or even maximum possible return) is specified. After

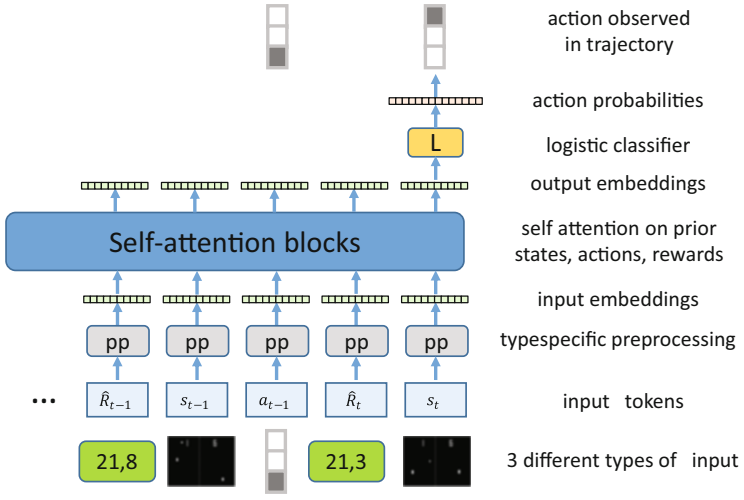


**Fig. 7.31** The Decision Transformer applies an autoregressive language model to the forward sums of rewards $\hat{R}_t$, states $s_t$ and actions $a_t$. In the example the state is given in the form of video frames, e.g. for the Pong game. The model predicts the next action in the trajectory conditional to a given forward sums of rewards [23]

the generated action $a_1$ is executed, the target return is reduced by the achieved reward and the next state $s_2$ is determined. This process of generating actions and applying them to get the next forward sum of rewards and the next state is repeated until the trajectory ends. Note that the actual forward sum of rewards should be close to the desired performance specified before. Although the model is only trained on randomly selected subsequences, it can learn to 'merge' subsequences from different training trajectories in order to produce optimal trajectories at test time. Obviously a large set of subsequences has to evaluated during training to arrive at good solutions.

The *Atari benchmark* [13] has discrete actions, uses four video frames as state descriptions, and processes these frames by a convolutional encoder. Only 1% of the available data is used. On four Atari tasks (Breakout, Qbert, Pong, and Seaquest) usually a context length of $K = 30$ is taken into account. With the exception of Qbert, Decision Transformer is competitive with state of the art methods, and for two games it reaches the best results (Breakout, Seaquest). The most effective alternative is the *CQL* [87] Q-learner.

The *D4RL benchmark* simulates simple robots (HalfCheetah, Hopper, and Walker) which are controlled by continuous-valued actions. On this benchmark Decision transformer in most cases achieves better results than the alternative approaches and has the highest average performance. Again CQL is the best alternative.

The authors evaluate the performance of approaches for an environment, where it is necessary to propagate rewards over a long time period. The *Key-to-Door benchmark* [104] has three phases:

- in the first phase, the agent is placed in a room with a key;
- then, the agent is placed in an empty room;
- and finally, the agent is placed in a room with a door.

The agent receives a binary reward when reaching the door in the third phase, but only if he picked up the key in the first phase. On this benchmark Decision Transformer and related methods clearly outperform Q-learning approaches, which cannot effectively propagate rewards over a long horizon.

Reid et al. [136] modify the details of the decision transformer yielding improved performance. Kumar et al. [86] show by theoretical analysis that offline reinforcement learning—as done by the decision transformer—enjoys better guarantees on long-horizon tasks than simply cloning the behavior of experts. This especially holds in the case of sufficiently noisy data.

### *7.4.2 The GATO Model for Text, Images and Control*

**GATO** [134] is a Foundation Model, which has been trained on about 600 different tasks, including text generation, image captioning, stacking physical blocks with a robot arm and playing Atari console games. Depending on the context, it

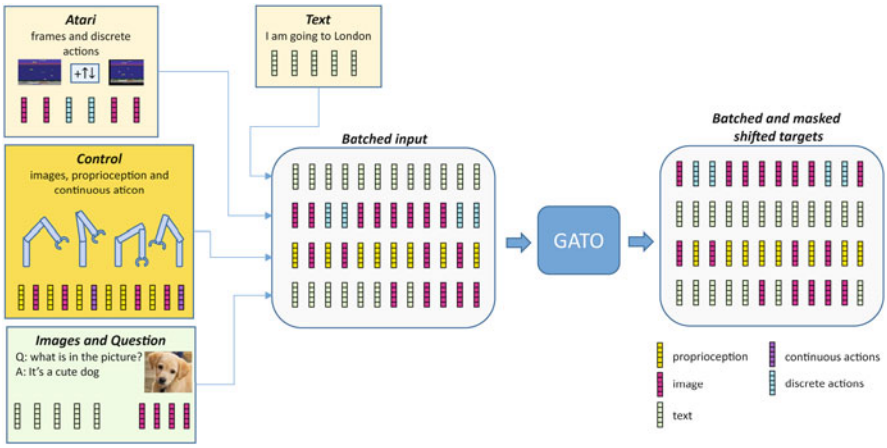


**Fig. 7.32** Data from different tasks and modalities are converted to sequences, e.g. frames and actions from Atari games, text token sequences, images patch tokens, continuous sensory inputs and outputs. In Gato [134, 135], a large decoder-only transformer processes the sequence. During training, specific variables, e.g. actions, are used to compute a loss. Image adapted from [135, fig.2], credits in Table A.3

independently decides which tokens to generate: Text, torques for joints, keystrokes, or another variant of the output within its comparatively extensive possibilities.

Depending on the modality the input is tokenized

- Text is encoded via SentencePiece with 32,000 tokens.
- Images are transformed into sequences of non-overlapping $16 \times 16$ images patches similar to the vision transformer (Sect. 7.2.2).
- Discrete values, e.g. Atari button presses, are flattened into sequences of integers in row-major order. The tokenized result is a sequence of integers within the range of [0, 1024].
- Continuous values, e.g. proprioceptive inputs (sense of self-movement, force, and body position) or joint torques, are preprocessed and discretized in 1024 bins. The discrete integers are then shifted to the range of [32,000, 33,024].

Tokens belonging to text, discrete- or continuous-valued observations, or actions for any time step are embedded into a learned vector embedding space using a lookup table. Learned position encodings are added for all tokens based on their local token position within their corresponding time step. Tokens belonging to image patches for any time step are embedded using a single ResNet [63] block to obtain a vector per image patch. In addition, a learnable within-image position encoding vector is added (Fig. 7.32).

Gato consists of a 1.2B parameter decoder-only transformer with 24 layers, and an embedding size of 2048. As in every language model, all tokens are predicted and therefore can be set as targets for training. Currently, only text tokens, discrete

and continuous values, and actions are currently used as targets. As usual, the probabilities of the observed target tokens have to be maximized during training.

To focus GATO on a specific task, a prompt is used coming from a trajectory generated by the same source agent on the same task. GATO was trained on 596 different control tasks, among them the Atari benchmark [13]. The authors included only "good" trajectories that yield at least 80% of the expert reward for the task. Moreover, GATO was trained on 8 vision and language tasks, e.g. image captioning with MS-COCO Captions [26] and Conceptual Captions [153], as well as visual question-answering datasets. In addition, GATO is trained on the large MassiveText [128] data with 300 billion text tokens.

The performance of GATO has been evaluated for different applications. On the Atari benchmark, the model reached average human score or better for 23 of 51 Atari games. In a robot stacking benchmark, GATO achieved a comparable performance as the BC-IMP baseline [90]. The model has only rudimentary dialog and caption functions, which is not surprising due to the small model size.

The Gato model is a first attempt to simultaneously solve text, image, and control tasks with the same Foundation Model. For control tasks it yielded respectable results while for the text and image tasks it had only mediocre performance. Perhaps it could benefit from the forward sum of rewards representation of the Decision Transformer. Actual Foundation Models have hundreds of billions of parameters and require a corresponding computing effort. If the GATO model is extended to this order of magnitude, its performance can be expected to improve accordingly.

**Available Implementations**

- Decision Transformer code https://sites.google.com/berkeley.edu/decision-transformer

### *7.4.3 Summary*

Pre-trained language models can be applied to sequences with mixtures of element types. The Decision Transformer considers sequences of rewards, states and actions at specific time steps, which occur during a sequential decision problem, e.g. video game playing, robot control, or automatic driving. It models observed trajectories of these quantities. Instead of using the reward as input, the sum of the rewards up to the end of the trajectory is considered, which is the quantity to be maximized. For each type of input some preprocessing is performed to generate embeddings. The Decision Transformer is trained to predict the actions in short subsequences of 30 time steps.

During application, the desired forward sum of rewards can be set as a condition. Then, the model is able to stitch together the information from different subsequences in the training data to obtain near-optimal actions reaching a maximal sum of rewards. This was shown by extensive experiments with various benchmarks.

The GATO model demonstrates that PLMs at the same time can be used to solve reinforcement learning tasks simultaneously with text and image tasks. The model is trained with nearly 600 control benchmarks, 8 image tasks and on 300B text tokens. The model has only rudimentary text and image description capabilities, but performs relatively well on the Atari benchmark. It is only a proof of concept and could be improved by increasing the model size and, for instance, by using the forward sum of rewards.

## 7.5 Interpretation of DNA and Protein Sequences

Deciphering the language of DNA is one of the most important goals of biological research. The genetic code is universal and explains how DNA is translated into proteins. In contrast, the regulatory code, which determines when and how genes are expressed, varies between different cell types and organisms. This is similar to the polysemy and distant semantic relationships in natural language texts. **DNABERT** [76] tokenizes the DNA sequence into overlapping 3-grams and trains a standard BERT model to predict masked tokens (Fig. 7.33). After pre-training on a large set of DNA sequences, it can improve the SOTA by fine-tuning for many specific DNA prediction tasks. Among them are analysis of sequence motifs (DNA segments with biological relevance) and prediction of promoter regions (nucleotide sequence that enables regulated expression of a gene). MoDNA [5] and GeneBERT [106] have similar functionality.

Proteins are linear chains of amino acids linked by covalent bonds. Amino acids can be represented by an alphabet with 25 characters. The strings are ideally suited for many NLP methods [111]. **AminoBERT** [29] is a language model that predicts

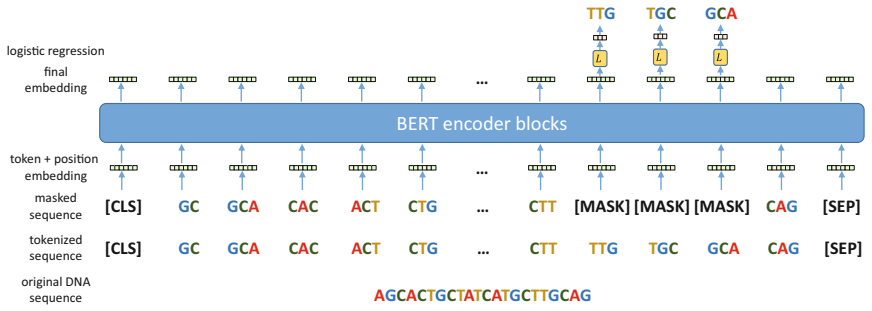


**Fig. 7.33** DNABERT tokenizes the DNA sequence into overlapping 3-grams and trains a standard BERT model to predict masked tokens [76]. The resulting model can be fine-tuned to many DNA interpretation tasks

the 3D protein structure given a protein sequence as input. It also uses a natural method to describe polypeptide geometry that is rotation and translation invariant at the level of the polypeptide as a whole. On average, the model outperforms AlphaFold2 [80] and RoseTTAFold [8] on orphan proteins and classes of engineered proteins, achieving up to a 106-fold reduction in computational time.

There are a number of other models with similar results [97], e.g., the protein language model **ESMFold**. It generates embeddings that can be used in downstream tasks, for example to capture the structural properties of proteins. A model with 15B parameters can predict the three-dimensional structure of a protein at the resolution of individual atoms.

**Available Implementations**

- DNABERT code and models https://github.com/jerryji1993/DNABERT
- GeneBERT code and models https://github.com/ZovcIfzm/GeneBERT/tree/main/GeneBERT
- ProteinBERT code and models https://github.com/nadavbra/protein_bert
- AlphaFold 2 code and models https://github.com/deepmind/alphafold
- RoseTTAFold code and models https://github.com/RosettaCommons/RoseTTAFold
- ESMFold code and models https://github.com/facebookresearch/esm

### *7.5.1 Summary*

Foundation Models can also be applied to DNA and protein sequences to derive contextual embeddings of the sequence elements. By this approach, the models are able to accumulate much knowledge about these sequences and achieve SOTA performance across various downstream tasks, largely surpassing existing tools. The models can help to predict the 3-D structure of the protein. This is crucial for its function and may be instrumental in developing active substances to influence it.

## References

1. T. Afouras, J. S. Chung, and A. Zisserman. “LRS3-TED: A Large-Scale Dataset for Visual Speech Recognition”. 2018. arXiv: 1809.00496.
2. H. Akbari, L. Yuan, R. Qian, W.-H. Chuang, S.-F. Chang, Y. Cui, and B. Gong. “VATT: Transformers for Multimodal Self-Supervised Learning from Raw Video, Audio and Text”. Dec. 6, 2021. arXiv: 2104.11178 [cs, eess].
3. J.-B. Alayrac et al. *Flamingo: A Visual Language Model for Few-Shot Learning*. Apr. 29, 2022. DOI: https://doi.org/10.48550/arXiv.2204.14198. arXiv: 2204.14198 [cs].
4. P. Ammanabrolu and M. Riedl. “Learning Knowledge Graph-Based World Models of Textual Environments”. In: *Adv. Neural Inf. Process. Syst.* 34 (2021), pp. 3720–3731.
5. W. An, Y. Guo, Y. Bian, H. Ma, J. Yang, C. Li, and J. Huang. “MoDNA: Motif-Oriented Pre-Training for DNA Language Model”. In: *Proc. 13th ACM Int. Conf. Bioinforma. Comput.*

*Biol. Health Inform.* BCB '22. New York, NY, USA: Association for Computing Machinery, Aug. 7, 2022, pp. 1–5. ISBN: 978-1-4503-9386-7. DOI: https://doi.org/10.1145/3535508.3545512.
6. P. Anderson. *VQA2VLN Tutorial 2021. From VQA to VLN: Recent Advances in Vision-and-Language Research*. June 20, 2021. URL: https://vqa2vln-tutorial.github.io/ (visited on 03/25/2022).
7. I. Anokhin, K. Demochkin, T. Khakhulin, G. Sterkin, V. Lempitsky, and D. Korzhenkov. "Image Generators with Conditionally-Independent Pixel Synthesis". In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2021, pp. 14278–14287.
8. M. Baek et al. "Accurate Prediction of Protein Structures and Interactions Using a Three-Track Neural Network". In: *Science* 373.6557 (Aug. 20, 2021), pp. 871–876. DOI: https://doi.org/10.1126/science.abj8754.
9. A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. "Data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language". Jan. 22, 2022. arXiv: 2202.03555.
10. A. Baevski, H. Zhou, A. Mohamed, and M. Auli. "Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations". 2020. arXiv: 2006.11477.
11. H. Bao, L. Dong, and F. Wei. "Beit: Bert Pre-Training of Image Transformers". 2021. arXiv: 2106.08254.
12. R. Beaumont. *LAION-5B: A NEW ERA OF OPEN LARGE-SCALE MULTI-MODAL DATASETS | LAION*. Aug. 8, 2022. URL: https://laion.ai/blog/laion-5b (visited on 08/29/2022).
13. M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. "The Arcade Learning Environment: An Evaluation Platform for General Agents". In: *J. Artif. Intell. Res.* 47 (2013), pp. 253–279.
14. Ş. Bilici. *A Survey On Music Generation With Deep Neural Networks*. Safak's Blog. Oct. 15, 2020. URL: https://safakkbilici.github.io/a-survey-on-music-generation/ (visited on 03/03/2022).
15. A. Blattmann, R. Rombach, K. Oktay, and B. Ommer. *Retrieval-Augmented Diffusion Models*. Apr. 26, 2022. DOI: https://doi.org/10.48550/arXiv.2204.11824. arXiv: 2204.11824 [cs].
16. A. Brock, S. De, S. L. Smith, and K. Simonyan. "High-Performance Large-Scale Image Recognition Without Normalization". 2021. arXiv: 2102.06171.
17. S. Cable. "Alexa, Read Me This Book in My Grandmother's Voice". In: *news* (June 24, 2022). ISSN: 0140-0460. URL: https://www.thetimes.co.uk/article/alexa-read-me-this-book-in-my-grandmothers-voice-cfdtjbjcc (visited on 07/08/2022).
18. R. Cai, J. Yuan, B. Xu, and Z. Hao. "SADGA: Structure-Aware Dual Graph Aggregation Network for Text-to-SQL". In: *Adv. Neural Inf. Process. Syst.* 34 (2021), pp. 7664–7676.
19. J. Cao, Z. Gan, Y. Cheng, L. Yu, Y.-C. Chen, and J. Liu. "Behind the Scene: Revealing the Secrets of Pre-Trained Vision-and-Language Models". In: *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 565–580.
20. Y.-H. Cao, H. Yu, and J. Wu. "Training Vision Transformers with Only 2040 Images". Jan. 25, 2022. arXiv: 2201.10728 [cs].
21. W. Chan, D. Park, C. Lee, Y. Zhang, Q. Le, and M. Norouzi. "Speechstew: Simply Mix All Available Speech Recognition Data to Train One Large Neural Network". 2021. arXiv: 2104.02133.
22. H. Chefer, S. Gur, and L. Wolf. "Transformer Interpretability beyond Attention Visualization". In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2021, pp. 782–791.
23. L. Chen et al. "Decision Transformer: Reinforcement Learning via Sequence Modeling". In: *Adv. Neural Inf. Process. Syst.* 34 (2021).
24. S.-J. Chen, A. S. Subramanian, H. Xu, and S. Watanabe. "Building State-of-the-Art Distant Speech Recognition Using the CHiME-4 Challenge with a Setup of Speech Enhancement Baseline". 2018. arXiv: 1803.10109.
25. W. Chen, M.-W. Chang, E. Schlinger, W. Wang, and W. W. Cohen. "Open Question Answering over Tables and Text". 2020. arXiv: 2010.10439.

26. X. Chen, H. Fang, T.-Y. Lin, R. Vedantam, S. Gupta, P. Dollár, and C. L. Zitnick. "Microsoft Coco Captions: Data Collection and Evaluation Server". 2015. arXiv: 1504.00325.
27. R. Child, S. Gray, A. Radford, and I. Sutskever. "Generating Long Sequences with Sparse Transformers". 2019. arXiv: 1904.10509.
28. J. Cho, J. Lu, D. Schwenk, H. Hajishirzi, and A. Kembhavi. "X-LXMERT: Paint, Caption and Answer Questions with Multi-Modal Transformers". 2020. arXiv: 2009.11278.
29. R. Chowdhury, N. Bouatta, and S. Biswas. "Single-Sequence Protein Structure Prediction Using a Language Model and Deep Learning". In: *Nat. Biotechnol.* (Oct. 3, 2022), pp. 1–7. URL: https://www.nature.com/articles/s41587-022-01432-w (visited on 10/14/2022).
30. Y.-S. Chuang, C.-L. Liu, H.-Y. Lee, and L.-s. Lee. "SpeechBERT: An Audio-and-text Jointly Learned Language Model for End-to-end Spoken Question Answering". Aug. 11, 2020. arXiv: 1910.11559 [cs, eess].
31. Y.-A. Chung, Y. Zhang, W. Han, C.-C. Chiu, J. Qin, R. Pang, and Y. Wu. "W2v-Bert: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training". 2021. arXiv: 2108.06209.
32. coco. *Papers with Code - COCO Captions Benchmark (Image Captioning)*. Mar. 6, 2022. URL: https://paperswithcode.com/sota/image-captioning-on-coco-captions (visited on 03/06/2022).
33. D. Damen et al. "Rescaling Egocentric Vision: Collection, Pipeline and Challenges for EPIC-KITCHENS-100". In: *Int. J. Comput. Vis.* 130.1 (2022), pp. 33–55.
34. K. Desai, G. Kaul, Z. Aysola, and J. Johnson. "RedCaps: Web-curated Image-Text Data Created by the People, for the People". 2021. arXiv: 2111.11431.
35. P. Dhariwal. *OpenAI Jukebox Sample Explorer*. 2020. URL: https://jukebox.openai.com/ (visited on 03/03/2022).
36. P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever. "Jukebox: A Generative Model for Music". Apr. 30, 2020. arXiv: 2005.00341 [cs, eess, stat].
37. P. Dhariwal and A. Nichol. "Diffusion Models Beat Gans on Image Synthesis". In: *Adv. Neural Inf. Process. Syst.* 34 (2021).
38. S. Di et al. "Video Background Music Generation with Controllable Music Transformer". In: *Proc. 29th ACM Int. Conf. Multimed.* 2021, pp. 2037–2045.
39. D. Ding, F. Hill, A. Santoro, M. Reynolds, and M. Botvinick. "Attention over Learned Object Embeddings Enables Complex Visual Reasoning". In: *Adv. Neural Inf. Process. Syst.* 34 (2021).
40. M. Ding et al. "CogView: Mastering Text-to-Image Generation via Transformers". Nov. 5, 2021. arXiv: 2105.13290 [cs].
41. A. Dosovitskiy and T. Brox. "Generating Images with Perceptual Similarity Metrics Based on Deep Networks". In: *Adv. Neural Inf. Process. Syst.* 29 (2016).
42. A. Dosovitskiy et al. "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". 2020. arXiv: 2010.11929.
43. Y. Du, Z. Liu, J. Li, and W. X. Zhao. "A Survey of Vision-Language Pre-Trained Models". 2022. arXiv: 2202.10936.
44. F. Ebert, C. Finn, A. X. Lee, and S. Levine. "Self-Supervised Visual Planning with Temporal Skip Connections." In: *CoRL*. 2017, pp. 344–356.
45. P. Esser, R. Rombach, and B. Ommer. "Taming Transformers for High-Resolution Image Synthesis". In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2021, pp. 12873–12883.
46. N. Fei et al. "WenLan 2.0: Make AI Imagine via a Multimodal Foundation Model". 2021. arXiv: 2110.14378.
47. W. Feller. "On the Theory of Stochastic Processes, with Particular Reference to Applications". In: *Proc. First Berkeley Symp. Math. Stat. Probab.* University of California Press, 1949, pp. 403–432.
48. O. Gafni, A. Polyak, O. Ashual, S. Sheynin, D. Parikh, and Y. Taigman. *Greater creative control for AI image generation*. July 14, 2022. URL: https://ai.facebook.com/blog/greater-creative-control-for-ai-image-generation/ (visited on 07/29/2022).

49. L. Gao et al. “The Pile: An 800GB Dataset of Diverse Text for Language Modeling”. 2020. arXiv: 2101.00027.
50. K. Gavrilyuk, R. Sanford, M. Javan, and C. G. Snoek. “Actor-Transformers for Group Activity Recognition”. In: Proc. IEEECVF *Conf. Comput. Vis. Pattern Recognit.* 2020, pp. 839–848.
51. S. Ging, M. Zolfaghari, H. Pirsiavash, and T. Brox. “COOT: Cooperative Hierarchical Transformer for Video-Text Representation Learning”. Nov. 1, 2020. arXiv: 2011.00597.
52. R. Girdhar, M. Singh, N. Ravi, L. van der Maaten, A. Joulin, and I. Misra. “Omnivore: A Single Model for Many Visual Modalities”. 2022. arXiv: 2201.08377.
53. I. Goodfellow et al. “Generative Adversarial Nets”. In: *Adv. Neural Inf. Process. Syst.* 2014, pp. 2672–2680.
54. google. *AVA: A Video Dataset of Atomic Visual Action*. 2020. URL: https://research.google.com/ava/ (visited on 03/12/2022).
55. R. Goyal et al. “The” Something Something” Video Database for Learning and Evaluating Visual Common Sense”. In: *Proc. IEEE Int. Conf. Comput. Vis.* 2017, pp. 5842–5850.
56. Y. Goyal, T. Khot, D. Summers-Stay, D. Batra, and D. Parikh. “Making the v in Vqa Matter: Elevating the Role of Image Understanding in Visual Question Answering”. In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2017, pp. 6904–6913.
57. A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. “Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks”. In: *Proc. 23rd Int. Conf. Mach. Learn.* 2006, pp. 369–376.
58. A. Gu, K. Goel, and C. Ré. “Efficiently Modeling Long Sequences with Structured State Spaces”. 2021. arXiv: 2111.00396.
59. A. Gulati et al. “Conformer: Convolution-augmented Transformer for Speech Recognition”. 2020. arXiv: 2005.08100.
60. Y. Guo et al. “From General to Specific: Informative Scene Graph Generation via Balance Adjustment”. In: *Proc. IEEECVF Int. Conf. Comput. Vis.* 2021, pp. 16383–16392.
61. K. Gupta, J. Lazarow, A. Achille, L. S. Davis, V. Mahadevan, and A. Shrivastava. “Layouttransformer: Layout Generation and Completion with Self-Attention”. In: *Proc. IEEECVF Int. Conf. Comput. Vis.* 2021, pp. 1004–1014.
62. A. M. Hafiz, S. A. Parah, and R. U. A. Bhat. “Attention Mechanisms and Deep Learning for Machine Vision: A Survey of the State of the Art”. June 3, 2021. arXiv: 2106.07550.
63. K. He, X. Zhang, S. Ren, and J. Sun. “Deep Residual Learning for Image Recognition”. In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2016, pp. 770–778.
64. M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. “Gans Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *Adv. Neural Inf. Process. Syst.* 30 (2017).
65. J. Ho, A. Jain, and P. Abbeel. “Denoising Diffusion Probabilistic Models”. In: *Adv. Neural Inf. Process. Syst.* 33 (2020), pp. 6840–6851.
66. J. Ho et al. “Imagen Video: High Definition Video Generation with Diffusion Models”. 2022. arXiv: 2210.02303.
67. M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga. “A Comprehensive Survey of Deep Learning for Image Captioning”. In: *ACM Comput. Surv. CsUR* 51.6 (2019), pp. 1–36.
68. W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed. “Hubert: Self-supervised Speech Representation Learning by Masked Prediction of Hidden Units”. In: *IEEEACM Trans. Audio Speech Lang. Process.* 29 (2021), pp. 3451–3460.
69. X. Hu, X. Yin, K. Lin, L. Wang, L. Zhang, J. Gao, and Z. Liu. “Vivo: Surpassing Human Performance in Novel Object Captioning with Visual Vocabulary Pre-Training”. 2020. arXiv: 2009.13682.
70. X. Hu, X. Yin, K. Lin, L. Zhang, J. Gao, L. Wang, and Z. Liu. “VIVO: Visual Vocabulary Pre-Training for Novel Object Captioning”. In: *Proc. AAAI Conf. Artif. Intell.* Vol. 35. 2. 2021, pp. 1575–1583.
71. C.-Z. A. Huang et al. “Music Transformer: Generating Music with Long-Term Structure”. In: *Int. Conf. Learn. Represent*. ICLR. 2019.

72. Y. Huang, H. Xue, B. Liu, and Y. Lu. "Unifying Multimodal Transformer for Bi-Directional Image and Text Generation". In: *Proc. 29th ACM Int. Conf. Multimed.* 2021, pp. 1138–1147.
73. S. Islam, A. Dash, A. Seum, A. H. Raj, T. Hossain, and F. M. Shah. "Exploring Video Captioning Techniques: A Comprehensive Survey on Deep Learning Methods". In: *SN Comput. Sci.* 2.2 (2021), pp. 1–28.
74. K. Ito and L. Johnson. *The LJ Speech Dataset.* 2017. URL: https://keithito.com/LJ-Speech-Dataset (visited on 03/24/2022).
75. E. Jang, S. Gu, and B. Poole. "Categorical Reparameterization with Gumbel-Softmax". 2016. arXiv: 1611.01144.
76. Y. Ji, Z. Zhou, H. Liu, and R. V. Davuluri. "DNABERT: Pre-Trained Bidirectional Encoder Representations from Transformers Model for DNA-language in Genome". In: *Bioinformatics* 37.15 (2021), pp. 2112–2120.
77. C. Jia and Y. Yang. *ALIGN: Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision*. Google AI Blog. May 11, 2021. URL: http://ai.googleblog.com/2021/05/align-scaling-up-visual-and-vision.html (visited on 06/08/2021).
78. Y. Jia. High-Quality, *Robust and Responsible Direct Speech-to-Speech Translation*. Google AI Blog. Sept. 23, 2021. URL: http://ai.googleblog.com/2021/09/high-quality-robust-and-responsible.html (visited on 10/25/2021).
79. D. Jin, Z. Jin, and R. Mihalcea. "Deep Learning for Text Attribute Transfer: A Survey". 2020. arXiv: 2011.00416.
80. J. Jumper et al. "Highly Accurate Protein Structure Prediction with AlphaFold". In: *Nature* 596.7873 (7873 Aug. 2021), pp. 583–589. issn: 1476-4687. DOI: https://doi.org/10.1038/s41586-021-03819-2.
81. T. Kano, S. Sakti, and S. Nakamura. "Transformer-Based Direct Speech-to-Speech Translation with Transcoder". In: (2021).
82. T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila. "Analyzing and Improving the Image Quality of Stylegan". In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2020, pp. 8110–8119.
83. W. Kay et al. "The Kinetics Human Action Video Dataset". 2017. arXiv: 1705.06950.
84. S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah. "Transformers in Vision: A Survey". In: *ACM Comput. Surv.* (Jan. 6, 2022), p. 3505244. issn: 0360-0300, 1557-7341. DOI: https://doi.org/10.1145/3505244.
85. K. Khurana and U. Deshpande. "Video Question-Answering Techniques, Benchmark Datasets and Evaluation Metrics Leveraging Video Captioning: A Comprehensive Survey." In: *IEEE Access* (2021).
86. A. Kumar, J. Hong, A. Singh, and S. Levine. *When Should We Prefer Offline Reinforcement Learning Over Behavioral Cloning?* Apr. 12, 2022. arXiv: 2204.05618 [cs].
87. A. Kumar, A. Zhou, G. Tucker, and S. Levine. "Conservative Q-Learning for Offline Reinforcement Learning". In: *Adv. Neural Inf. Process. Syst.* 33 (2020), pp. 1179–1191.
88. M. Kumar, D. Weissenborn, and N. Kalchbrenner. "Colorization Transformer". 2021. arXiv: 2102.04432.
89. K. Lakhotia et al. "Generative Spoken Language Modeling from Raw Audio". Sept. 9, 2021. arXiv: 2102.01192 [cs].
90. A. X. Lee et al. "Beyond Pick-and-Place: Tackling Robotic Stacking of Diverse Shapes". In: *5th Annu. Conf. Robot Learn.* 2021.
91. H. Lee, U. Ullah, J.-S. Lee, B. Jeong, and H.-C. Choi. "A Brief Survey of Text Driven Image Generation and Maniulation". In: *2021 IEEE Int. Conf. Consum. Electron.-Asia ICCE-Asia*. IEEE, 2021, pp. 1–4.
92. Z. Leng, M. Tan, C. Liu, E. D. Cubuk, J. Shi, S. Cheng, and D. Anguelov. "PolyLoss: A Polynomial Expansion Perspective of Classification Loss Functions". In: *Int. Conf. Learn. Represent.* 2021.
93. M. Li et al. "CLIP-Event: Connecting Text and Images with Event Structures". 2022. arXiv: 2201.05078.

94. N. Li, S. Liu, Y. Liu, S. Zhao, and M. Liu. "Neural Speech Synthesis with Transformer Network". In: *Proc. AAAI Conf. Artif. Intell.* Vol. 33. 01. 2019, pp. 6706–6713.
95. X. Li et al. "Oscar: Object-semantics Aligned Pre-Training for Vision-Language Tasks". In: *Eur. Conf. Comput. Vis.* Springer, 2020, pp. 121–137.
96. J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte. "Swinir: Image Restoration Using Swin Transformer". In: *Proc. IEEECVF Int. Conf. Comput. Vis.* 2021, pp. 1833–1844.
97. Z. Lin et al. "Language Models of Protein Sequences at the Scale of Evolution Enable Accurate Structure Prediction". In: *bioRxiv* (2022).
98. A. T. Liu, S.-W. Li, and H.-y. Lee. "Tera: Self-supervised Learning of Transformer Encoder Representation for Speech". In: *IEEEACM Trans. Audio Speech Lang. Process.* 29 (2021), pp. 2351–2366.
99. Z. Liu et al. "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows". In: *Proc. IEEECVF Int. Conf. Comput. Vis.* 2021, pp. 10012–10022.
100. J. Lu, D. Batra, D. Parikh, and S. Lee. "Vilbert: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks". In: *Adv. Neural Inf. Process. Syst.* 2019, pp. 13–23.
101. M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom. "Automatic Speech Recognition: A Survey". In: *Multimed. Tools Appl.* (2020), pp. 1–47.
102. M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom. "Automatic Speech Recognition: A Survey". In: *Multimed. Tools Appl.* 80.6 (2021), pp. 9411–9457.
103. C. Mao, L. Jiang, M. Dehghani, C. Vondrick, R. Sukthankar, and I. Essa. "Discrete Representations Strengthen Vision Transformer Robustness". Nov. 19, 2021. arXiv: `2111.10493 [cs]`.
104. T. Mesnard et al. "Counterfactual Credit Assignment in Model-Free Reinforcement Learning". 2020. arXiv: `2011.09464`.
105. A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic. "HowTo100M: Learning a Text-Video Embedding by Watching Hundred Million Narrated Video Clips". July 31, 2019. arXiv: `1906.03327 [cs]`.
106. S. Mo et al. "Multi-Modal Self-supervised Pre-training for Regulatory Genome Across Cell Types". 2021. arXiv: `2110.05231`.
107. M. Monfort et al. "Moments in Time Dataset: One Million Videos for Event Understanding". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 42.2 (2019), pp. 502–508.
108. M. M. Naseer, K. Ranasinghe, S. H. Khan, M. Hayat, F. Shahbaz Khan, and M.-H. Yang. "Intriguing Properties of Vision Transformers". In: *Adv. Neural Inf. Process. Syst.* 34 (2021).
109. A. Nichol et al. "Glide: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models". 2021. arXiv: `2112.10741`.
110. A. Q. Nichol and P. Dhariwal. "Improved Denoising Diffusion Probabilistic Models". In: *Int. Conf. Mach. Learn.* PMLR, 2021, pp. 8162–8171.
111. D. Ofer, N. Brandes, and M. Linial. "The Language of Proteins: NLP, Machine Learning & Protein Sequences". In: *Comput. Struct. Biotechnol. J.* 19 (2021), pp. 1750–1758.
112. A. Oluwasammi et al. "Features to Text: A Comprehensive Survey of Deep Learning on Semantic Segmentation and Image Captioning". In: *Complexity* 2021 (2021).
113. A. van den Oord, O. Vinyals, and K. Kavukcuoglu. "Neural Discrete Representation Learning". May 30, 2018. arXiv: `1711.00937 [cs]`.
114. A. van den Oord et al. "Wavenet: A Generative Model for Raw Audio". 2016. arXiv: *1609.03499*.
115. OpenAI. DALL·E Now Available in Beta. July 20, 2022. URL: https://openai.com/blog/dall-e-now-available-in-beta/ (visited on 07/29/2022).
116. V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. "Librispeech: An ASR Corpus Based on Public Domain Audio Books". In: *2015 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP*. IEEE, 2015, pp. 5206–5210.
117. I. Papastratis. *Speech Recognition: A Review of the Different Deep Learning Approaches*. AI Summer. July 14, 2021. URL: https://theaisummer.com/speech-recognition/ (visited on 03/02/2022).

118. papers-with-code. *Papers with Code - ImageNet Benchmark (Image Classification)*. 2022. URL: https://paperswithcode.com/sota/image-classification-on-imagenet (visited on 03/05/2022).
119. K. K. Parida, S. Srivastava, and G. Sharma. "Beyond Mono to Binaural: Generating Binaural Audio from Mono Audio with Depth and Cross Modal Attention". In: *Proc. IEEECVF Winter Conf. Appl. Comput. Vis.* 2022, pp. 3347–3356.
120. D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. "Specaugment: A Simple Data Augmentation Method for Automatic Speech Recognition". 2019. arXiv: 1904.08779.
121. D. S. Park et al. "Improved Noisy Student Training for Automatic Speech Recognition". 2020. arXiv: 2005.09629.
122. T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. "Semantic Image Synthesis with Spatially-Adaptive Normalization". Nov. 5, 2019. arXiv: 1903.07291 [cs].
123. C. Payne. "MuseNet". In: *OpenAI Blog* (2019).
124. J. Perez-Martin, B. Bustos, S. J. F. Guimarães, I. Sipiran, J. Pérez, and G. C. Said. "Bridging Vision and Language from the Video-to-Text Perspective: A Comprehensive Review". 2021. arXiv: 2103.14785.
125. R. Prenger, R. Valle, and B. Catanzaro. "Waveglow: A Flow-Based Generative Network for Speech Synthesis". In: *ICASSP 2019-2019 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP*. IEEE, 2019, pp. 3617–3621.
126. A. Radford, I. Sutskever, J. W. Kim, G. Krueger, and S. Agarwal. *CLIP: Connecting Text and Images*. Jan. 5, 2021. URL: https://openai.com/blog/clip/.
127. A. Radford et al. "Learning Transferable Visual Models from Natural Language Supervision". In: *Int. Conf. Mach. Learn.* PMLR, 2021, pp. 8748–8763.
128. J. W. Rae et al. "Scaling Language Models: Methods, Analysis & Insights from Training Gopher". In: *ArXiv Prepr. ArXiv211211446* (Dec. 8, 2021), p. 118.
129. c. raffel. *C4 | TensorFlow Datasets*. TensorFlow. 2019. URL: https://www.tensorflow.org/datasets/catalog/c4 (visited on 12/14/2021).
130. M. Raghu, T. Unterthiner, S. Kornblith, C. Zhang, and A. Dosovitskiy. "Do Vision Transformers See Like Convolutional Neural Networks?" In: (Dec. 1, 2021), p. 13.
131. P. Ramachandran, B. Zoph, and Q. V. Le. "Searching for Activation Functions". 2017. arXiv: 1710.05941.
132. A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. "Hierarchical Text-Conditional Image Generation with CLIP Latents". Apr. 12, 2022. arXiv: 2204.06125 [cs].
133. A. Ramesh et al. "Zero-Shot Text-to-Image Generation". Feb. 26, 2021. arXiv: 2102.12092.
134. S. Reed. A Generalist Agent. May 12, 2022. URL: https://www.deepmind.com/publications/a-generalist-agent (visited on 05/19/2022).
135. S. Reed et al. A Generalist Agent. May 12, 2022. arXiv: 2205.06175 [cs].
136. M. Reid, Y. Yamada, and S. S. Gu. "Can Wikipedia Help Offline Reinforcement Learning?" Jan. 28, 2022. arXiv: 2201.12122 [cs].
137. S. Ren, K. He, R. Girshick, and J. Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". Jan. 6, 2016. arXiv: 1506.01497 [cs].
138. Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T.-Y. Liu. "FastSpeech 2: Fast and High-Quality End-to-End Text to Speech". Mar. 4, 2021. arXiv: 2006.04558.
139. M. Rivière and E. Dupoux. "Towards Unsupervised Learning of Speech Features in the Wild". In: *2021 IEEE Spok. Lang. Technol. Workshop SLT*. IEEE, 2021, pp. 156–163.
140. J. Rodriguez. *Five Key Facts Wu Dao 2.0: The Largest Transformer Model Ever Built*. DataSeries. Sept. 21, 2021. URL: https://medium.com/dataseries/five-key-facts-wu-dao-2-0-the-largest-transformer-model-ever-built-19316159796b (visited on 12/12/2021).
141. R. Rombach. *Latent Diffusion Models*. CompVis - Machine Vision and Learning LMU Munich, Aug. 29, 2022. URL: https://github.com/CompVis/latent-diffusion (visited on 08/29/2022).

142. R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. *High Resolution Image Synthesiss with Latent Diffusion...* -. CVPR 22. Apr. 13, 2022. URL: https://scholar.google.com/scholar?hl=de&as_sdt=0%2C5&q=high+resolution+image+synthesiss+with+latent+diffusion+models&btnG= (visited on 08/29/2022).
143. A. Romero. *GPT-3 Scared You? Meet Wu Dao 2.0: A Monster of 1.75 Trillion Parameters*. Medium. June 8, 2021. URL: https://towardsdatascience.com/gpt-3-scared-you-meet-wu-dao-2-0-a-monster-of-1-75-trillion-parameters-832cd83db484 (visited on 07/29/2021).
144. O. Ronneberger, P. Fischer, and T. Brox. "U-Net: Convolutional Networks for Biomed- ical Image Segmentation". In: *Int. Conf. Med. Image Comput. Comput.-Assist. Interv.* Springer, 2015, pp. 234–241.
145. L. Ruan and Q. Jin. "Survey: Transformer Based Video-Language Pre-Training". In: *AI Open* 3 (Jan. 1, 2022), pp. 1–13. issn: 2666-6510. DOI: https://doi.org/10.1016/j.aiopen.2022.01.001.
146. M. S. Ryoo, A. J. Piergiovanni, A. Arnab, M. Dehghani, and A. Angelova. "TokenLearner: What Can 8 Learned Tokens Do for Images and Videos?" 2021. arXiv: 2106.11297.
147. C. Saharia, W. Chan, and S. Saxena. Imagen: Text-to-Image Diffusion Models. May 25, 2022. URL: https://imagen.research.google/ (visited on 05/26/2022).
148. C. Saharia et al. "Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding". May 23, 2022. DOI: https://doi.org/10.48550/arXiv.2205.11487. arXiv: 2205.11487 [cs].
149. I. Salian. *NVIDIA Research's GauGAN AI Art Demo Responds to Words*. NVIDIA Blog. Nov. 22, 2021. URL: https://blogs.nvidia.com/blog/2021/11/22/gaugan2-ai-art-demo/ (visited on 03/06/2022).
150. T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. "Improved Techniques for Training Gans". In: *Adv. Neural Inf. Process. Syst.* 29 (2016).
151. C. Schuhmann. *LAION-400-Million Open Dataset*. LAION. Aug. 20, 2021. URL: https://laion.ai/laion-400-open-dataset/ (visited on 03/05/2022).
152. D. Serdyuk, O. Braga, and O. Siohan. "Transformer-Based Video Front-Ends for Audio-Visual Speech Recognition". 2022. arXiv: 2201.10439.
153. P. Sharma, N. Ding, S. Goodman, and R. Soricut. "Conceptual Captions: A Cleaned, Hypernymed, Image Alt-Text Dataset for Automatic Image Captioning". In: *Proc. 56th Annu. Meet. Assoc. Comput. Linguist. Vol. 1 Long Pap.* 2018, pp. 2556–2565.
154. J. Shen et al. "Natural Tts Synthesis by Conditioning Wavenet on Mel Spectrogram Predictions". In: *2018 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP*. IEEE, 2018, pp. 4779–4783.
155. Y.-J. Shih, S.-L. Wu, F. Zalkow, M. Müller, and Y.-H. Yang. "Theme Transformer: Symbolic Music Generation with Theme-Conditioned Transformer". Nov. 7, 2021. arXiv: 2111.04093 [cs, eess].
156. J. Shor. *TRILLsson: Small, Universal Speech Representations for Paralinguistic Tasks*. Google AI Blog. Mar. 3, 2022. URL: http://ai.googleblog.com/2022/03/trillsson-small-universal-speech.html (visited on 03/29/2022).
157. J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. "Deep Unsupervised Learning Using Nonequilibrium Thermodynamics". In: *Int. Conf. Mach. Learn.* PMLR, 2015, pp. 2256–2265.
158. Stable. *Stable Diffusion Online*. 2022. URL: https://stablediffusionweb.com/ (visited on 12/31/2022).
159. M. Stefanini, M. Cornia, L. Baraldi, S. Cascianelli, G. Fiameni, and R. Cucchiara. "From Show to Tell: A Survey on Image Captioning". 2021. arXiv: 2107.06912.
160. C. Sun, A. Myers, C. Vondrick, K. Murphy, and C. Schmid. "Videobert: A Joint Model for Video and Language Representation Learning". In: *Proc. IEEE Int. Conf. Comput. Vis.* 2019, pp. 7464–7473.
161. C. Sun, A. Shrivastava, S. Singh, and A. Gupta. "Revisiting Unreasonable Effectiveness of Data in Deep Learning Era". In: *Proc. IEEE Int. Conf. Comput. Vis.* 2017, pp. 843–852.

162. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. "Rethinking the Inception Architecture for Computer Vision". In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2016, pp. 2818–2826.
163. X. Tan, T. Qin, F. Soong, and T.-Y. Liu. "A Survey on Neural Speech Synthesis". July 23, 2021. arXiv: 2106.15561.
164. Z. Tang, J. Lei, and M. Bansal. "Decembert: Learning from Noisy Instructional Videos via Dense Captions and Entropy Minimization". In: *Proc. 2021 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol.* 2021, pp. 2415–2426.
165. M. Tao, H. Tang, S. Wu, N. Sebe, X.-Y. Jing, F. Wu, and B. Bao. "DF-GAN: Deep Fusion Generative Adversarial Networks for Text-to-Image Synthesis". Mar. 24, 2021. arXiv: 2008.05865.
166. M. Tsimpoukelli, J. L. Menick, S. Cabi, S. M. Eslami, O. Vinyals, and F. Hill. "Multimodal Few-Shot Learning with Frozen Language Models". In: *Adv. Neural Inf. Process. Syst.* 34 (2021), pp. 200–212.
167. T. Unterthiner, S. van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly. "Towards Accurate Generative Models of Video: A New Metric & Challenges". 2018. arXiv: 1812.01717.
168. A. Vaswani et al. "Attention Is All You Need". In: *Adv. Neural Inf. Process. Syst.* 2017, pp. 5998–6008.
169. R. Vedantam, C. Lawrence Zitnick, and D. Parikh. "Cider: Consensus-based Image Description Evaluation". In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2015, pp. 4566–4575.
170. P. Wang et al. "OFA: Unifying Architectures, Tasks, and Modalities Through a Simple Sequence-to-Sequence Learning Framework". 2022. arXiv: 2202.03052.
171. Z. Wang, J. Yu, A. W. Yu, Z. Dai, Y. Tsvetkov, and Y. Cao. "SimVLM: Simple Visual Language Model Pretraining with Weak Supervision". Aug. 24, 2021. arXiv: 2108.10904.
172. D. Weissenborn, O. Täckström, and J. Uszkoreit. "Scaling Autoregressive Video Models". In: *ICLR* (2020).
173. C.-Y. Wu, Y. Li, K. Mangalam, H. Fan, B. Xiong, J. Malik, and C. Feichtenhofer. "MeMViT: Memory-Augmented Multiscale Vision Transformer for Efficient Long-Term Video Recognition". 2022. arXiv: 2201.08383.
174. C. Wu. *Overview.* Microsoft, Mar. 14, 2022. URL: https://github.com/microsoft/NUWA (visited on 03/14/2022).
175. C. Wu, J. Liang, L. Ji, F. Yang, Y. Fang, D. Jiang, and N. Duan. "Nüwa: Visual Synthesis Pre-Training for Neural Visual World Creation". 2021. arXiv: 2111.12417.
176. C. Wu et al. "Godiva: Generating Open-Domain Videos from Natural Descriptions". 2021. arXiv: 2104.14806.
177. Z. Wu, D. Lischinski, and E. Shechtman. "StyleSpace Analysis: Disentangled Controls for StyleGAN Image Generation". Dec. 3, 2020. arXiv: 2011.12799 [cs].
178. N. Xie, F. Lai, D. Doran, and A. Kadav. "Visual Entailment: A Novel Task for Fine-Grained Image Understanding". 2019. arXiv: 1901.06706.
179. S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. "Aggregated Residual Transformations for Deep Neural Networks". In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2017, pp. 1492–1500.
180. S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. "Rethinking Spatiotemporal Feature Learning for Video Understanding". 2017. arXiv: 1712.04851.
181. W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. "The Microsoft 2017 Conversational Speech Recognition System". In: *2018 IEEE Int. Conf. Acoust. Speech Signal Process. ICASSP*. IEEE, 2018, pp. 5934–5938.
182. J. Xu, T. Mei, T. Yao, and Y. Rui. "Msr-Vtt: A Large Video Description Dataset for Bridging Video and Language". In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2016, pp. 5288–5296.
183. P. Xu, X. Zhu, and D. A. Clifton. *Multimodal Learning with Transformers: A Survey*. June 13, 2022. DOI: https://doi.org/10.48550/arXiv.2206.06488. arXiv: 2206.06488 [cs].

184. Q. Xu et al. “Self-Training and Pre-training Are Complementary for Speech Recognition”. 2021. arXiv: 2010.11430.
185. S. Yan, X. Xiong, A. Arnab, Z. Lu, M. Zhang, C. Sun, and C. Schmid. “Multiview Transformers for Video Recognition”. In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2022, pp. 3333–3343.
186. Y. Yan, X. Tan, B. Li, T. Qin, S. Zhao, Y. Shen, and T.-Y. Liu. “AdaSpeech 2: Adaptive Text to Speech with Untranscribed Data”. Apr. 19, 2021. arXiv: 2104.09715 [cs, eess].
187. L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg. “Modeling Context in Referring Expressions”. In: *Eur. Conf. Comput. Vis.* Springer, 2016, pp. 69–85.
188. R. Zellers et al. “Merlot: Multimodal Neural Script Knowledge Models”. In: *Adv. Neural Inf. Process. Syst.* 34 (2021).
189. X. Zhai, A. Kolesnikov, N. Houlsby, and L. Beyer. “Scaling Vision Transformers”. June 8, 2021. arXiv: 2106.04560 [cs].
190. B. Zhang, J. Yu, C. Fifty, W. Han, A. M. Dai, R. Pang, and F. Sha. “Co-Training Transformer with Videos and Images Improves Action Recognition”. Dec. 14, 2021. arXiv: 2112.07175 [cs].
191. B. Zhang et al. “StyleSwin: Transformer-based GAN for High-resolution Image Generation”. 2021. arXiv: 2112.10762.
192. H. Zhang, J. Y. Koh, J. Baldridge, H. Lee, and Y. Yang. “Cross-Modal Contrastive Learning for Text-to-Image Generation”. 2021. arXiv: 2101.04702.
193. P. Zhang et al. “VinVL: Making Visual Representations Matter in Vision-Language Models”. 2021. arXiv: 2101.00529.
194. R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* 2018, pp. 586–595.
195. Y. Zhang et al. “BigSSL: Exploring the Frontier of Large-Scale Semi-Supervised Learning for Automatic Speech Recognition”. Oct. 1, 2021. arXiv: 2109.13226 [cs, eess].
196. Y. Zhang et al. “Pushing the Limits of Semi-Supervised Learning for Automatic Speech Recognition”. 2020. arXiv: 2010.10504.
197. L. Zhao, D. Cai, L. Sheng, and D. Xu. “3DVG-Transformer: Relation Modeling for Visual Grounding on Point Clouds”. In: *Proc. IEEECVF Int. Conf. Comput. Vis.* 2021, pp. 2928–2937.
198. A. Zhavoronkov. *Wu Dao 2.0 - Bigger, Stronger, Faster AI From China*. Forbes. July 19, 2021. URL: https://www.forbes.com/sites/alexzhavoronkov/2021/07/19/wu-dao-20bigger-stronger-faster-ai-from-china/ (visited on 07/29/2021).
199. H. Zhou, W. Zhou, W. Qi, J. Pu, and H. Li. “Improving Sign Language Translation with Monolingual Data by Sign Back-Translation”. In: *Proc. IEEECVF Conf. Comput. Vis. Pattern Recognit.* 2021, pp. 1316–1325.
200. Y. Zhou et al. “LAFITE: Towards Language-Free Training for Text-to-Image Generation”. 2021. arXiv: 2111.13792.
201. X. Zhu et al. “Multi-Modal Knowledge Graph Construction and Application: A Survey”. 2022. arXiv: 2202.05786.
202. D. Zügner, T. Kirschstein, M. Catasta, J. Leskovec, and S. Günnemann. “Language-Agnostic Representation Learning of Source Code from Structure and Context”. 2021. arXiv: 2103.11318.