

# Natural Language Processing (NLP)

- “Efficient Estimation of Word Representations in Vector Space” (Tomas Mikolov *et al.*, 2013) – [paper](#). Introduces **word2vec** (skip-gram and CBOW) to learn continuous word embeddings. Word2vec efficiently produces high-quality word vectors that capture precise syntactic and semantic relationships <sup>1</sup>. (Code: [Mikolov's word2vec](#)).
- “Sequence to Sequence Learning with Neural Networks” (Ilya Sutskever *et al.*, 2014) – [paper](#). First to show an end-to-end **LSTM encoder-decoder** that maps input sequences to output sequences. Achieved strong machine translation results (34.8 BLEU on English–French) by training two LSTMs jointly <sup>2</sup>. (Code: e.g. [TensorFlow seq2seq example](#)).
- “Neural Machine Translation by Jointly Learning to Align and Translate” (Dzmitry Bahdanau *et al.*, 2015) – [paper](#). Introduces the **attention mechanism** for NMT. By allowing the decoder to “attend” to relevant encoder states dynamically, it overcomes the fixed-size context bottleneck. The model achieved competitive translation performance and produces interpretable soft-alignments <sup>3</sup>. (Code: [Bahdanau's NMT code](#)).
- “Attention Is All You Need” (Vaswani *et al.*, 2017) – [paper](#). Proposes the **Transformer** architecture using only self-attention (no recurrent or convolutional layers). Transformers parallelize better and learn dependencies more directly. Vaswani et al. demonstrate state-of-the-art machine translation and faster training (BLEU 28.4 on EN–DE) with fully attention-based models <sup>4</sup>. (Code: [Tensor2Tensor](#) / [OpenNMT](#)).
- “Deep Contextualized Word Representations” (**ELMo**) (Peters *et al.*, 2018) – [paper](#). Introduces **ELMo**, contextual word embeddings from a biLSTM language model. ELMo captures word meaning from context (syntax/semantics), improving diverse tasks (QA, NLI, sentiment) by large margins <sup>5</sup>. (Code: [AllenNLP ELMo](#)).
- “Universal Language Model Fine-Tuning for Text Classification” (**ULMFiT**) (Howard & Ruder, 2018) – [paper](#). Demonstrates that **transfer learning** (pre-train on large corpus, then fine-tune on task) works for NLP. ULMFiT applies discriminative fine-tuning and slanted triangular learning rates to LSTM language models. It obtains huge error reductions (18–24%) on multiple text classification benchmarks <sup>6</sup>. (Code: [stprarakis/ulmfit](#)).
- “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding” (Devlin *et al.*, 2018) – [paper](#). Introduces **BERT**, a deep Transformer model pre-trained on masked language modeling and next-sentence prediction. BERT’s bidirectional context and fine-tuning scheme yield state-of-the-art results on many benchmarks (GLUE, SQuAD) with minimal architecture changes <sup>7</sup>. (Code: [google-research/bert](#) <sup>8</sup>).
- “Improving Language Understanding by Generative Pre-Training” (**GPT**) (Radford *et al.*, 2018) – [paper](#). Presents **GPT** (OpenAI’s first large Transformer LM). Shows that unsupervised pre-training on diverse text, followed by discriminative fine-tuning, yields strong gains on multiple NLP tasks. GPT (117M parameters) outperforms many baselines on 9 of 12 NLP benchmarks without heavy task-specific architecture <sup>9</sup>. (Code: [openai/finetune-transformer-lm](#)).
- “Language Models are Unsupervised Multitask Learners” (**GPT-2**) (Radford *et al.*, 2019) – [paper](#). Introduces **GPT-2**, a 1.5B-parameter Transformer. Without fine-tuning, GPT-2 performs many tasks (QA, translation, summarization) in a zero/few-shot manner by conditioning on textual prompts. The model matches or exceeds SOTA on 7/8 language benchmarks and can generate highly coherent paragraphs <sup>10</sup> <sup>11</sup>. (Code: [openai/gpt-2](#)).

- “**RoBERTa: A Robustly Optimized BERT Pretraining Approach**” (Liu *et al.*, 2019) – [paper](#). Shows BERT can be improved by training longer/bigger on more data with hyperparameter tweaks. RoBERTa removes the next-sentence objective, uses dynamic masking, larger batches, and larger corpora. The result matches or exceeds all models since BERT: RoBERTa’s best model sets SOTA on GLUE, RACE, and SQuAD <sup>12</sup>. (Code: [facebookresearch/fairseq](#) includes RoBERTa).
- “**XLNet: Generalized Autoregressive Pretraining for Language Understanding**” (Yang *et al.*, 2019) – [paper](#). Introduces **XLNet**, which combines autoregressive and autoencoding: it maximizes likelihood over permutations of input tokens (via Transformer-XL backbone). XLNet captures bidirectional context without masking. It significantly outperforms BERT on 20+ tasks, notably QA and NLI <sup>13</sup> <sup>14</sup>. (Code: [CMU-Kit/XLNet](#)).
- “**ALBERT: A Lite BERT for Self-supervised Learning of Language Representations**” (Lan *et al.*, 2019) – [paper](#). Proposes **ALBERT**, which shares parameters across layers and factorizes embeddings to reduce model size. Also introduces a sentence-order prediction loss. Despite far fewer parameters, ALBERT matches or exceeds BERT-large: 33x fewer parameters achieve new SOTA on GLUE/RACE/SQuAD <sup>15</sup> <sup>16</sup>. (Code: [google-research/albert](#)).
- “**BART: Denoising Sequence-to-Sequence Pre-training for NLG, Translation, and Comprehension**” (Lewis *et al.*, 2019) – [paper](#). Introduces **BART**, a seq2seq Transformer pretrained as a denoising autoencoder (corrupt text with noise, reconstruct). BART combines a bidirectional encoder (like BERT) and autoregressive decoder (like GPT). When fine-tuned, it matches RoBERTa on language understanding and achieves new SOTA on generation tasks (summarization, dialogue, QA) <sup>17</sup> <sup>18</sup>. (Code: [facebookresearch/fairseq](#) includes BART).
- “**Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer**” (T5) (Raffel *et al.*, 2020) – [paper](#). Casts every NLP task as text generation (“text-to-text”) and scales a Transformer up to 11B parameters. Trained on massive cleaned web crawl (C4), **T5** achieves SOTA results on a wide range of tasks (summarization, QA, translation, etc.) by simply framing tasks as text transformation <sup>19</sup> <sup>20</sup>. (Code: [google-research/text-to-text-transfer-transformer](#)).
- “**ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators**” (Clark *et al.*, 2020) – [paper](#). Proposes **ELECTRA**, a more sample-efficient pretraining: a small generator replaces tokens (like masked LM), and a larger discriminator must detect replaced tokens. This uses *all* tokens for learning. ELECTRA trains faster: small ELECTRA models outperform much larger BERT/GPT with the same compute <sup>21</sup> <sup>22</sup>. (Code: [google-research/electra](#)).
- “**Dense Passage Retrieval for Open-Domain Question Answering**” (DPR) (Karpukhin *et al.*, 2020) – [paper](#). Proposes **DPR**, a dual-encoder dense retrieval approach for open-domain QA. Questions and passages are embedded with BERT encoders, and nearest-neighbor retrieval finds relevant docs. DPR outperforms BM25 by 9–19% in top-20 recall, enabling state-of-the-art QA when combined with a reader model <sup>23</sup>. (Code: [facebookresearch/DPR](#)).
- “**Retrieval-Augmented Generation (RAG)**” (Lewis *et al.*, 2020) – [paper](#). Combines generative seq2seq models with a non-parametric retriever over a knowledge corpus. RAG fetches relevant documents for each query and conditions generation on them. Two variants (RAG-Sequence and RAG-Token) achieve SOTA on 3 open-domain QA tasks, surpassing both purely neural models and pipeline methods. RAG produces more factual answers by referencing retrieved passages <sup>24</sup> <sup>25</sup>. (Code: [huggingface/transformers RAG examples](#)).
- “**SpanBERT: Improving Pre-training by Representing and Predicting Spans**” (Joshi *et al.*, 2020) – [paper](#). Extends BERT by masking contiguous spans (not just single tokens) and training to predict entire spans. SpanBERT also trains span-boundary representations. This yields large gains on span-based tasks: SOTA on QA (SQuAD F1 94.6) and coreference (OntoNotes F1 79.6) <sup>26</sup>. (Code: [facebookresearch/SpanBERT](#)).

- “**DeBERTa: Decoding-enhanced BERT with Disentangled Attention**” (He *et al.*, 2021) – [paper](#). Proposes two improvements over BERT/RoBERTa: **disentangled attention** (separate content and position embeddings) and an enhanced mask decoder. Combined with adversarial fine-tuning, **DeBERTa** achieves higher efficiency and performance: e.g. a 1.5B-parameter DeBERTa surpasses RoBERTa-large and even reaches human-level (89.9% SuperGLUE) <sup>27</sup> <sup>28</sup>. (Code: [microsoft/DeBERTa](#)).
- “**Longformer: The Long-Document Transformer**” (Beltagy *et al.*, 2020) – [paper](#). Introduces **Longformer**, a Transformer for long texts using sliding-window (and optional global) attention. It scales linearly in sequence length, allowing modeling thousands of tokens. Pretrained Longformer outperforms RoBERTa on long-context tasks and achieves new SOTA on benchmarks like WikiHop and TriviaQA. The authors also present **LED**, a Longformer encoder-decoder for long-document summarization <sup>29</sup> <sup>30</sup>. (Code: [allenai/longformer](#)).
- “**Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity**” (Fedus *et al.*, 2021) – [paper](#). Proposes **Switch Transformer**, a Mixture-of-Experts (MoE) model that routes tokens to a subset of experts (sparsity). Switch reduces training cost (up to 7× speedup) by activating only one expert per token. The paper demonstrates scalable training of trillion-parameter LMs on natural and multilingual corpora, with simpler routing and stable training at low precision <sup>31</sup> <sup>32</sup>. (Code: [google/mesh-transformer-jax](#) supports Switch).

## Computer Vision (CV)

- “**Gradient-Based Learning Applied to Document Recognition**” (**LeNet-5**) (LeCun *et al.*, 1998) – [paper](#). Introduces **LeNet-5**, one of the first convolutional neural networks (CNNs), for handwritten digit recognition. LeNet-5 used convolution, subsampling, and fully connected layers, demonstrating CNNs’ power. (Code: [LeNet-5 TensorFlow example](#)).
- “**ImageNet Classification with Deep Convolutional Neural Networks**” (**AlexNet**) (Krizhevsky *et al.*, 2012) – [paper](#). Achieved a breakthrough on ImageNet (AlexNet): an 8-layer CNN that won the 2012 ILSVRC by a large margin (15.3% top-5 error vs. next best 26.2%). AlexNet showed deep CNNs (ReLUs, dropout, GPU training) could learn powerful image classifiers. (Code: [cuda-connet](#)).
- “**Very Deep Convolutional Networks for Large-Scale Image Recognition**” (**VGG**) (Simonyan & Zisserman, 2014) – [paper](#). Demonstrated that stacking many small (3×3) convolution filters can greatly improve classification accuracy. VGG-16/VGG-19 (16/19 layers) achieved 7.3% top-5 error on ImageNet. The paper also provided pre-trained deep features still widely used. (Code: [keras/applications/VGG16](#)).
- “**Going Deeper with Convolutions**” (**Inception-v1**) (Szegedy *et al.*, 2015) – [paper](#). Introduces the **Inception (GoogLeNet)** architecture, which uses parallel “Inception” modules (multi-branch convolutions and pooling) to increase network width. The 22-layer model won ILSVRC 2014 (6.7% error) with relatively low parameters. (Code: [tensorflow/models/inception](#)).
- “**Deep Residual Learning for Image Recognition**” (**ResNet**) (He *et al.*, 2015) – [paper](#). Proposes **ResNets** with skip (residual) connections to ease training of very deep nets (50–152 layers). ResNet won ILSVRC 2015 (3.57% error) and became a backbone for most CV tasks. Residual connections enable learning of identity mappings, alleviating vanishing gradients and yielding better accuracy <sup>4</sup>. (Code: [keras/applications/ResNet50](#)).
- “**Faster R-CNN: Towards Real-Time Object Detection**” (Ren *et al.*, 2015) – [paper](#). Introduces **Faster R-CNN**, unifying region proposal and classification in one network with a Region Proposal Network (RPN). It runs faster than previous R-CNNs and achieves SOTA in object detection on PASCAL/VOC and COCO. (Code: [facebookresearch/detectron2](#) includes Faster R-CNN).

- “**You Only Look Once (YOLO)**” (Redmon *et al.*, 2016) – [paper](#). Proposes **YOLO**, a single-shot real-time object detector that divides the image into a grid and predicts bounding boxes and class probabilities in one pass. YOLOv1 runs at 45 FPS (fast) with competitive mAP. Subsequent YOLO versions (v2–v5) further improve speed/accuracy. (Code: [pjreddie/darknet](#)).
- “**Densely Connected Convolutional Networks (DenseNet)**” (Huang *et al.*, 2017) – [paper](#). Introduces DenseNets, where each layer receives inputs from all previous layers (concatenation). This encourages feature reuse and alleviates vanishing gradients. DenseNets achieve comparable accuracy to ResNets with fewer parameters on ImageNet and CIFAR <sup>33</sup> . (Code: [github.com/liuzhuang13/DenseNet](#)).
- “**Feature Pyramid Networks (FPN) for Object Detection**” (Lin *et al.*, 2017) – [paper](#). Introduces FPN, a top-down architecture that builds a multi-scale feature pyramid from a single-resolution input. By combining coarse, semantic high-level features with finer low-level ones, FPN greatly improves detection of objects at different scales. Used in Mask R-CNN, RetinaNet, etc. (Code: [Detectron2 FPN](#)).
- “**Mask R-CNN**” (He *et al.*, 2017) – [paper](#). Extends Faster R-CNN by adding a branch for pixel-level **mask prediction**, enabling simultaneous object detection and instance segmentation. Mask R-CNN is simple (adds only a small FCN on top of RoI features) but achieves state-of-art instance segmentation on COCO. (Code: [Detectron2 Mask R-CNN](#)).
- “**Squeeze-and-Excitation Networks**” (SENet) (Hu *et al.*, 2017) – [paper](#). Proposes the **SE block**, which adaptively recalibrates channel-wise feature responses by modeling interdependencies. Adding SE blocks to ResNet/VGG improves ImageNet classification accuracy with marginal extra cost. SE won the 2017 ILSVRC team prize. (Code: [keras-squeeze-excite-network](#)).
- “**Path Aggregation Network (PANet)**” (Liu *et al.*, 2018) – [paper](#). Builds on FPN by adding bottom-up path augmentation and adaptive feature pooling, improving information flow for instance segmentation. PANet was part of the winning COCO 2017 segmentation entry, yielding stronger feature hierarchies for Mask R-CNN. (Code: [Detectron2 PANet](#)).
- “**YOLOv4: Optimal Speed and Accuracy of Object Detection**” (Bochkovskiy *et al.*, 2020) – [paper](#). Introduces YOLOv4, which integrates many “bag of freebies” and “bag of specials” (CSPDarknet backbone, mish activation, mosaic augmentation, SATM, etc.) to achieve 65.7% AP on COCO at 65 FPS on a Tesla V100. It significantly improved speed-accuracy tradeoff over prior YOLO versions. (Code: [AlexeyAB/darknet](#)).
- “**EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks**” (Tan & Le, 2019) – [paper](#). Proposes a principled scaling method using a compound coefficient to uniformly scale depth, width, and resolution. **EfficientNet** (eight models B0–B7) achieves state-of-the-art ImageNet accuracy (up to 85.3% top-1) with an order of magnitude fewer parameters than previous models <sup>33</sup> . (Code: [tensorflow/models/official/efficientnet](#)).
- “**Vision Transformer (ViT)**” (Dosovitskiy *et al.*, 2020) – [paper](#). Applies a pure Transformer (no convolutions) to images by splitting them into patches (16×16 tokens). ViT, pretrained on large datasets (ImageNet-21k or JFT-300M), matches or exceeds CNNs on image classification and transfer tasks. It shows Transformers can serve as competitive vision backbones. (Code: [google-research/vision\\_transformer](#)).
- “**An Image is Worth 16×16 Words: Transformers for Image Recognition at Scale**” (ViT) (same as above).
- “**DETR: End-to-End Object Detection with Transformers**” (Carion *et al.*, 2020) – [paper](#). Introduces **DETR**, which formulates object detection as direct set prediction with a Transformer encoder-decoder and bipartite matching loss. DETR removes the need for anchors or NMS, simplifying the detection pipeline. While slower to train, DETR achieves competitive COCO detection results in an elegant end-to-end manner. (Code: [facebookresearch/detr](#)).

- “**Deformable DETR**” (Zhu *et al.*, 2020) – [paper](#). Improves DETR by using deformable attention to focus on sparse, relevant spatial locations. Deformable DETR trains much faster (few epochs vs 500) and handles objects of varied scales better, achieving strong detection performance with reduced computational cost. (Code: [facebookresearch/detectron2](#)).
- “**Swin Transformer: Hierarchical Vision Transformer using Shifted Windows**” (Liu *et al.*, 2021) – [paper](#). Proposes **Swin Transformer**, a hierarchical Transformer with shifted non-overlapping windows. It computes attention within local windows but shifts them between layers to capture cross-window connections. Swin achieves SOTA across image classification, detection, and segmentation (80.2% top-1 on ImageNet-1K with 3.0B MAdds) <sup>34</sup>. (Code: [microsoft/Swin-Transformer](#)).
- “**Self-Supervised Learning of Visual Representations by Contrasting Cluster Assignments**” (DINO) (Caron *et al.*, 2021) – [paper](#). Demonstrates that **self-distillation** (DINO) can train ViT models without labels. DINO’s attention maps align with object parts. The resulting ViTs achieve SOTA unsupervised features for segmentation/detection. (Code: [facebookresearch/dino](#)).
- “**Segment Anything Model (SAM)**” (Kirillov *et al.*, 2023) – [paper](#). Introduces **SAM**, a promptable segmentation model trained on a colossal dataset (1B masks, 11M images). SAM generalizes to any segmentation prompt (points, boxes, masks) and achieves strong zero-shot results on diverse data, often matching fully supervised methods <sup>35</sup>. It is a foundation model for segmentation tasks. (Code: [facebookresearch/segment-anything](#)).

## Reinforcement Learning (RL)

- “**Playing Atari with Deep Reinforcement Learning**” (DQN) (Mnih *et al.*, 2015) – [paper](#). Pioneering **Deep Q-Network (DQN)**, which used a CNN to approximate Q-values from raw pixels to play Atari games. DQN achieved human-level control on many games. It introduced experience replay and target networks to stabilize Q-learning. (Code: [Google DQN in TensorFlow](#)).
- “**Deep Reinforcement Learning with Double Q-learning**” (van Hasselt *et al.*, 2016) – [paper](#). Identifies DQN’s tendency to overestimate values. **Double DQN** uses two networks to decouple action selection and evaluation, reducing overestimation. This yields more stable learning and higher scores on Atari <sup>36</sup> <sup>37</sup>. (Code: [Double DQN](#)).
- “**Dueling Network Architectures for Deep RL**” (Wang *et al.*, 2016) – [paper](#). Proposes the **dueling architecture**, which separates state-value and advantage estimation in the network. This factorization helps learn which states are valuable regardless of action, improving learning when many actions yield similar outcomes. The resulting agent outperforms DQN on Atari <sup>38</sup>. (Code: [OpenAI Baselines](#) includes dueling DQN).
- “**Trust Region Policy Optimization (TRPO)**” (Schulman *et al.*, 2015) – [paper](#). Introduces **TRPO**, a policy gradient method with a trust region constraint to ensure monotonic improvement. TRPO reliably optimizes deep policy networks across continuous and discrete tasks (simulated locomotion, Atari) without extensive tuning, setting new baselines. (Code: [OpenAI baselines TRPO](#)).
- “**Continuous control with Deep Reinforcement Learning**” (DDPG) (Lillicrap *et al.*, 2016) – [paper](#). Extends DQN ideas to continuous action spaces via **Deep Deterministic Policy Gradient (DDPG)**. DDPG is an off-policy actor-critic with a deterministic policy and target networks, enabling learning of continuous control (Mujoco locomotion/manipulation) with sample efficiency competitive with model-based methods <sup>39</sup>. (Code: [Stable Baselines DDPG](#)).
- “**Asynchronous Methods for Deep RL**” (A3C) (Mnih *et al.*, 2016) – [paper](#). Proposes **A3C**, which runs multiple actor-learner threads in parallel with shared parameters. This decorrelates updates and speeds up training without GPUs. A3C achieved state-of-the-art results on Atari (half the training

time on CPU) and succeeded on continuous control and 3D navigation tasks <sup>40</sup>. (Code: [OpenAI baselines A2C/A3C](#)).

- “**Proximal Policy Optimization (PPO)**” (Schulman *et al.*, 2017) – [paper](#). Presents **PPO**, a simpler alternative to TRPO using a clipped surrogate objective. PPO avoids large policy updates by clipping probability ratios. It achieves strong performance on Atari and MuJoCo with only first-order updates, combining sample efficiency and simplicity <sup>41</sup>. (Code: [OpenAI baselines PPO](#)).
- “**Rainbow: Combining Improvements in Deep RL**” (Hessel *et al.*, 2018) – [paper](#). Introduces **Rainbow DQN**, an agent that integrates six DQN extensions (Double DQN, Dueling networks, Multi-step returns, Prioritized replay, Noisy nets, Distributional RL). Rainbow achieves state-of-art sample efficiency and high scores on Atari by combining these enhancements <sup>42</sup>. (Code: [third-party Rainbow](#)).
- “**Soft Actor-Critic (SAC)**” (Haarnoja *et al.*, 2018) – [paper](#). Introduces **SAC**, an off-policy actor-critic that maximizes a tradeoff between return and policy entropy (maximum entropy RL). This encourages exploration and stability. SAC yields state-of-the-art performance on continuous control benchmarks (e.g. outperforms PPO/DDPG) and is more sample-efficient <sup>43</sup>. (Code: [stable-baselines3 SAC](#)).
- “**Addressing Function Approximation Error in Actor-Critic Methods**” (**TD3**) (Fujimoto *et al.*, 2018) – [paper](#). Identifies overestimation bias in actor-critic methods. **TD3** (Twin Delayed DDPG) uses clipped double Q-learning (take min of two critics) and delayed policy updates to reduce error. TD3 consistently outperforms DDPG and prior methods on continuous control (OpenAI Gym) <sup>44</sup>. (Code: [stable-baselines3 TD3](#)).
- “**IMPALA: Scalable Distributed RL with Actor-Learner Architectures**” (Espeholt *et al.*, 2018) – [paper](#). Proposes **IMPALA**, a distributed actor-critic framework with many workers generating experience in parallel, and a central learner. Uses *V*-trace off-policy correction to handle lag. IMPALA scales to 1000s of cores and demonstrates strong multi-task transfer on 3D environments (e.g. DMLab, Atari) <sup>45</sup>. (Code: [OpenAI baselines](#) includes IMPALA).
- “**MuZero**” (Schrittwieser *et al.*, 2020) – [paper](#). Combines model-based planning with learning: MuZero learns a model of dynamics purely from reward/predictions (no game rules). It then uses Monte Carlo Tree Search on the learned model. MuZero achieves SOTA on Atari and matches AlphaZero on Go, chess, and shogi, without access to game rules <sup>46</sup>. (Code: [Pytorch MuZero implementation](#)).
- “**Decision Transformer**” (Chen *et al.*, 2021) – [paper](#). Recasts RL as sequence modeling: a GPT-like transformer is conditioned on desired returns and past states to generate actions. Surprisingly, the **Decision Transformer** matches or exceeds specialized offline RL methods on Atari and Gym without explicit reward or policy learning, by learning from off-policy trajectories <sup>47</sup> <sup>48</sup>. (Code: [jarobin/decision-transformer](#)).
- “**Conservative Q-Learning (CQL) for Offline RL**” (Kumar *et al.*, 2020) – [paper](#). Addresses offline (batch) RL where data is fixed. **CQL** adds a penalty to the Q-function so it lower-bounds the value of the behavior policy, preventing overestimation of unseen actions. CQL achieves much higher returns (2–5×) than prior offline RL on complex continuous tasks, enabling reliable learning from static datasets <sup>49</sup>. (Code: [ucbdrive/cql](#)).

## Diffusion Models

- “**Deep Unsupervised Learning using Nonequilibrium Thermodynamics**” (Sohl-Dickstein *et al.*, 2015) – [paper](#). Early formulation of **diffusion probabilistic models**. It defines a forward diffusion (adding noise) and a learnable reverse process to generate data. Although not as performant on

images, it laid the foundation for later diffusion models by treating generation as gradually denoising. (Historic reference; code often built on later improvements).

- “**Denoising Diffusion Probabilistic Models (DDPM)**” (Ho *et al.*, 2020) – [paper](#). Proposes the now-standard **DDPM** framework: a Markovian diffusion adds Gaussian noise for  $T$  steps, and a neural net (U-Net) is trained to reverse it. DDPMs generate high-quality images rivaling GANs, but require many timesteps to sample. (Code: [openai/improved-diffusion](#)).
- “**Score-Based Generative Modeling through SDEs**” (Song *et al.*, 2021) – [paper](#). Unifies diffusion and score-based models by defining continuous stochastic differential equations (SDEs) that convert data to noise and back. The reverse SDE depends on score (gradient of log-density). Using score networks and numerical SDE solvers, this method produces high-fidelity samples (e.g. 1024×1024 images) and enables likelihood computation. It achieves record CIFAR-10 scores (IS 9.89, FID 2.20) <sup>50</sup>. (Code: [ermongroup/ncsn](#)).
- “**Denoising Diffusion Implicit Models (DDIM)**” (Song *et al.*, 2020) – [paper](#). Shows that one can use a **non-Markovian diffusion** with the same training as DDPMs but an implicit sampling process. **DDIM** enables deterministic or accelerated sampling. Empirically, DDIMs generate high-quality images 10–50x faster (fewer steps) than DDPMs, with the ability to trade off speed vs quality <sup>51</sup>.
- “**Classifier-Free Diffusion Guidance**” (Ho & Salimans, 2022) – [paper](#). Introduces a simpler alternative to classifier guidance for conditional generation. By jointly training a diffusion model with and without conditioning, one can interpolate between conditional and unconditional scores at sampling time. **Classifier-free guidance** achieves the same high-fidelity sampling quality as classifier-based guidance (e.g. in GLIDE/Imagen) without a separate classifier <sup>52</sup>. (Code: integrated in [HuggingFace Diffusers](#)).
- “**Improved Denoising Diffusion Probabilistic Models**” (Nichol & Dhariwal, 2021) – [paper](#). Enhances DDPMs with new techniques: learning the variance schedule and adding conditioning. They show improved likelihoods and that sampling can use far fewer steps (by learning variances) with minimal loss in sample quality. The paper includes **guided diffusion** for class-conditional images, demonstrating diffusion models *beat GANs* on high-resolution synthesis <sup>53</sup>. (Code: [openai/improved-diffusion](#)).
- “**Diffusion Models Beat GANs on Image Synthesis**” (Dhariwal & Nichol, 2021) – [paper](#). Trains large-scale conditional diffusion models (guided by classifier) and achieves state-of-the-art on ImageNet. Demonstrates diffusion models can match or surpass GANs in sample quality when scaled, and that they are more stable to train. This work popularized diffusion as a top-generation method.
- “**Hierarchical Text-Conditional Image Generation with CLIP Latents (GLIDE)**” (Nichol *et al.*, 2021) – [paper](#). (Note: Paper uses diffusion for text-to-image.) Trains large diffusion models conditioned on CLIP text embeddings, allowing high-quality text-guided image synthesis (precedes DALL-E 2). Demonstrates that diffusion models can be guided by learned text encoders to generate semantically relevant images.
- “**Imagen: Text-to-Image Diffusion Models**” (Saharia *et al.*, 2022) – [paper](#). Combines diffusion with large pre-trained language (T5) for text conditioning. Imagen achieves unprecedented fidelity and accuracy in text-to-image generation, reaching human-level FID on COCO. The paper shows scaling up diffusion models and using powerful text encoders drastically improves alignment between text and images. (Code: not released, but similar to GLIDE guidance and HuggingFace).
- “**High-Resolution Image Synthesis with Latent Diffusion Models**” (Rombach *et al.*, 2022) – [paper](#). Introduces **Latent Diffusion Models (LDMs)**, which operate in the lower-dimensional latent space of an autoencoder. This makes high-resolution (e.g. 1024×1024) generation tractable on consumer GPUs. LDMs (e.g. Stable Diffusion) achieve image quality comparable to pixel-diffusion at a fraction of compute. (Code: [CompVis/stable-diffusion](#)).

# Generative Adversarial Networks (GANs)

- “**Generative Adversarial Networks**” (Goodfellow *et al.*, 2014) – [paper](#). Introduces the GAN framework: a **generator** network tries to produce realistic samples to fool a **discriminator** network, which learns to distinguish real vs. fake. The two play a minimax game. Remarkably, with sufficient capacity the optimal solution equates to the model replicating the data distribution <sup>54</sup>. GANs avoid explicit likelihoods and Markov chains, instead generating via neural nets.
- “**Conditional Generative Adversarial Nets**” (Mirza & Osindero, 2014) – [paper](#). Extends GANs by conditioning on labels or attributes. A **cGAN** generator and discriminator both receive class labels as input. Conditioning allows control over the generated content (e.g. digit class) and improves sample quality. (Code: [TensorFlow cGAN example](#)).
- “**Deep Convolutional Generative Adversarial Networks (DCGANs)**” (Radford *et al.*, 2016) – [paper](#). Applies convolutional networks to GANs. DCGAN imposes architectural constraints (no pooling, batch normalization, ReLU in generator, Leaky ReLU in discriminator). This stable architecture learns hierarchical image features (edges→objects) without supervision <sup>33</sup>. DCGAN marked the first visually convincing GANs on natural images. (Code: [dcgan.torch](#)).
- “**InfoGAN: Interpretable Representation Learning by Information Maximizing GANs**” (Chen *et al.*, 2016) – [paper](#). An information-theoretic GAN extension. **InfoGAN** maximizes mutual information between a subset of latent codes and generated outputs, encouraging **disentangled** representations. It discovers interpretable factors (e.g. digit class, rotation) in MNIST and CelebA without labels <sup>55</sup>. (Code: [TensorFlow InfoGAN](#)).
- “**Pix2Pix: Image-to-Image Translation with Conditional GANs**” (Isola *et al.*, 2017) – [paper](#). Demonstrates how cGANs can map between image domains (e.g. edges→photo, day→night) when paired training data is available. Pix2Pix uses a U-Net generator and PatchGAN discriminator, achieving photorealistic outputs. It popularized GANs for tasks like super-resolution and style transfer. (Code: [phillipi/pix2pix](#)).
- “**CycleGAN: Unpaired Image-to-Image Translation**” (Zhu *et al.*, 2017) – [paper](#). Extends Pix2Pix to *unpaired* data by using cycle-consistency: two generators translate in both directions and must reconstruct original images. **CycleGAN** successfully translates between domains (e.g. horses↔zebras) without paired examples. This enables many style-transfer applications. (Code: [junyanz/CycleGAN](#)).
- “**Progressive Growing of GANs**” (Karras *et al.*, 2018) – [paper](#). Introduces progressively training GANs, starting from low resolution and incrementally adding layers. This stabilizes high-resolution image generation. Progressive GAN produced high-fidelity celeb faces up to 1024×1024. (Code: [NVIDIA/progressive\\_growing\\_of\\_gans](#)).
- “**Spectral Normalization for GANs**” (Miyato *et al.*, 2018) – [paper](#). Introduces **spectral normalization**, a weight regularization enforcing Lipschitz constraint by normalizing each layer’s weight spectral norm. It stabilizes training of GANs (especially on large resolutions) and improves Inception Score/FID. (Code: [sn-gan](#)).
- “**Self-Attention Generative Adversarial Networks (SAGAN)**” (Zhang *et al.*, 2019) – [paper](#). Adds **self-attention** to GANs, allowing global dependencies across the image (instead of purely convolutional local). SAGAN improves fine-grained detail generation (e.g. object coherence across space) and sets new SOTA on ImageNet generation by attending over feature maps. (Code: [zhanghang1989/ResNeSt](#) includes SAGAN).
- “**BigGAN: Large Scale GAN Training for High Fidelity Natural Image Synthesis**” (Brock *et al.*, 2019) – [paper](#). Scales GANs up massively (up to 512×512 on ImageNet) by using very large batch sizes and class-conditional BigGANs with orthonormal initialization and truncated normal sampling.

BigGAN produces very high-fidelity images and demonstrates that large batch training can push GAN performance to SOTA. (Code: [deepmind/biggan-pytorch](#)).

- “**StyleGAN: A Style-Based Generator Architecture for Generative Adversarial Networks**” (Karras *et al.*, 2019) – [paper](#). Introduces **StyleGAN**, which maps latent vectors to style via adaptive instance normalization (AdaIN) at each convolutional layer. This enables control over coarse-to-fine features (big facial features vs. color). StyleGAN achieves state-of-the-art face generation, producing extremely realistic, disentangled images. (Code: [NVlabs/stylegan](#)).
- “**StyleGAN2: Analyzing and Improving the Image Quality of StyleGAN**” (Karras *et al.*, 2020) – [paper](#). Improves StyleGAN by removing artifacts (replacing AdaIN with weight demodulation, etc.) and updating normalization. StyleGAN2 produces even sharper and more stable images. It became a new standard for high-resolution face/scene synthesis. (Code: [NVlabs/stylegan2](#)).
- “**StyleGAN3: Alias-Free Generative Adversarial Networks**” (Karras *et al.*, 2021) – [paper](#). Further refines StyleGAN to remove translation/rotation artifacts (aliasing) by enforcing strict signal processing. StyleGAN3 produces samples that are truly consistent under shifts/rotations. (Code: [NVlabs/stylegan3](#)).
- “**StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation**” (Choi *et al.*, 2018) – [paper](#). Proposes **StarGAN**, which extends CycleGAN to multiple domains with a single model. By conditioning the generator on target domain labels, StarGAN can translate images across many domains (hair color, age, etc.) using one network. (Code: [yunjey/stargan](#)).
- “**Projection Discriminator in Conditional GANs**” (Miyato & Koyama, 2018) – [paper](#). Introduces the **projection discriminator** for conditional GANs. The discriminator computes inner products between embedded labels and features. This simple change yields large performance gains in class-conditional generation (applied in BigGAN). (Code: [tensorflow-GAN](#)).
- “**GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium**” (Heusel *et al.*, 2017). Theoretical analysis of GAN training, also introduces **TTUR** (different learning rates for G and D). (Less a code source, but a foundational understanding of GAN convergence.)

## Large Language Models (LLMs)

- “**Language Models are Few-Shot Learners**” (GPT-3) (Brown *et al.*, 2020) – [paper](#). Introduces **GPT-3**, a 175B-parameter autoregressive Transformer. Remarkably, GPT-3 can perform many NLP tasks (translation, Q&A, arithmetic, etc.) with *zero/few-shot prompting* (no gradient updates). It achieves state-of-the-art results on dozens of benchmarks by simply conditioning on textual instructions or examples [56](#) [57](#). (Code: model not public; similar GPT-3 models exist in Hugging Face).
- “**PaLM: Scaling Language Modeling with Pathways**” (Chowdhery *et al.*, 2022) – [paper](#). Presents **PaLM**, a 540B-parameter densely activated Transformer trained on massive data with Google’s Pathways. PaLM achieves new SOTA few-shot performance on hundreds of tasks, especially reasoning (mathematics, logic) and multilingual benchmarks, and outperforms previous best on BIG-bench by a large margin [34](#) [58](#). (Code: none public; related [t5x](#)).
- “**LLaMA: Open and Efficient Foundation Language Models**” (Touvron *et al.*, 2023) – [paper](#). Introduces **LLaMA**, a family of open-source LMs (7B–65B) trained on public data. Surprisingly, the 13B model *outperforms* GPT-3 (175B) on most benchmarks, and LLaMA-65B rivals the best models (Chinchilla-70B, PaLM-540B) [59](#). By releasing these models, LLaMA democratized research on large LMs. (Code/Weights: [Meta’s release](#)).
- “**Training Compute-Optimal Large Language Models**” (Chinchilla) (Hoffmann *et al.*, 2022) – [paper](#). Analyzes scaling laws and shows many LMs (GPT-3, Gopher) were undertrained. Proposes **Chinchilla**:

a 70B-parameter model trained with ~4x more data than a comparable Gopher. Chinchilla achieves state-of-the-art on many benchmarks (MMLU 67.5%) and outperforms much larger models (GPT-3 175B, Gopher 280B) while using less compute <sup>60</sup> <sup>61</sup>. (Code: none public; influences training of models like LLaMA).

- “**GLM-130B: General Language Model**” (Zeng *et al.*, 2022) – [paper](#). Introduces **GLM-130B**, a Chinese-English autoregressive + autoencoding LM trained on 1T tokens. It achieves strong bilingual performance and top few-shot Chinese results among open models. (Code: not public).
- “**Jurassic-1: At-Scale Autoregressive Language Models**” (Dai *et al.*, 2022) – [paper](#). (Preprint describes 178B-parameter model). Jurassic-1 is a collection of large English LMs (incl. 178B). It demonstrates strong few-shot learning and provides an API for multi-domain text tasks. (Code: none public; model via commercial API).
- “**Megatron-Turing NLG**” (**MT-NLG**) (Smith *et al.*, 2022) – [paper](#). A 530B-parameter LM by NVIDIA/Microsoft (combining Megatron and Turing-NLG). MT-NLG achieves SOTA on zero-shot NLU/NLG tasks (SuperGLUE, Winograd, LAMBADA) with 10–20% improvements. (Code: [NVIDIA Megatron-LM](#)).
- “**FLAN: Fine-tuned LMs Are Zero-shot Learners**” (Wei *et al.*, 2021) – [paper](#). Showed that instruction-tuning (fine-tuning LMs on a collection of tasks with instructions) dramatically improves zero-shot capabilities. The authors fine-tuned T5 on ~1.8K tasks (FLAN) and observed large gains in generalization to novel tasks. (Code: [google-research/FLAN](#)).
- “**UL2: Unifying Language Learning Paradigms**” (Tay *et al.*, 2022) – [paper](#). Proposes **UL2**, a new pretraining framework mixing denoising tasks (span-prediction, infilling, etc.) for sequence models. UL2 outperforms both decoder-only and encoder-decoder baselines on diverse tasks (SuperGLUE, AR summarization, etc.) with fewer parameters <sup>62</sup>. (Code: [google-research/ul2](#)).
- “**Galactica: Large Language Model for Science**” (Taylor *et al.*, 2022). A 120B-parameter LM trained on scientific papers, code, and data. Galactica generates scientific text and has strong reasoning over factual information. (Code: none public; demo online).
- “**StableLM**” (**Stability AI, 2023**) – A series of open LLMs (3B, 7B) fine-tuned with StableAI. (Website: [Stability AI](#)).

*Sources:* Key papers and their descriptions are drawn from their abstracts and introductions <sup>1</sup> <sup>9</sup> <sup>10</sup> <sup>21</sup>, with supplementary details from official code repos where noted. These citations provide authoritative context for each model’s contributions.

<sup>1</sup> [1310.4546] Distributed Representations of Words and Phrases and their Compositionality  
<https://arxiv.org/abs/1310.4546>

<sup>2</sup> [1409.3215] Sequence to Sequence Learning with Neural Networks  
<https://arxiv.org/abs/1409.3215>

<sup>3</sup> [1409.0473] Neural Machine Translation by Jointly Learning to Align and Translate  
<https://arxiv.org/abs/1409.0473>

<sup>4</sup> [1706.03762] Attention Is All You Need  
<https://arxiv.org/abs/1706.03762>

<sup>5</sup> [1802.05365] Deep contextualized word representations  
<https://arxiv.org/abs/1802.05365>

<sup>6</sup> [1801.06146] Universal Language Model Fine-tuning for Text Classification  
<https://arxiv.org/abs/1801.06146>

- 7 [1810.04805] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding  
<https://arxiv.org/abs/1810.04805>
- 8 GitHub - google-research/bert: TensorFlow code and pre-trained models for BERT  
<https://github.com/google-research/bert>
- 9 cdn.openai.com  
[https://cdn.openai.com/research-covers/language-unsupervised/language\\_understanding\\_paper.pdf](https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf)
- 10 11 Language Models are Unsupervised Multitask Learners  
[https://cdn.openai.com/better-language-models/language\\_models\\_are\\_unsupervised\\_multitask\\_learners.pdf](https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf)
- 12 [1907.11692] RoBERTa: A Robustly Optimized BERT Pretraining Approach  
<https://arxiv.org/abs/1907.11692>
- 13 14 [1906.08237] XLNet: Generalized Autoregressive Pretraining for Language Understanding  
<https://arxiv.org/abs/1906.08237>
- 15 16 [1909.11942] ALBERT: A Lite BERT for Self-supervised Learning of Language Representations  
<https://arxiv.org/abs/1909.11942>
- 17 18 [1910.13461] BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension  
<https://arxiv.org/abs/1910.13461>
- 19 20 [1910.10683] Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer  
<https://arxiv.org/abs/1910.10683>
- 21 22 [2003.10555] ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators  
<https://arxiv.org/abs/2003.10555>
- 23 [2004.04906] Dense Passage Retrieval for Open-Domain Question Answering  
<https://arxiv.org/abs/2004.04906>
- 24 25 [2005.11401] Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks  
<https://arxiv.org/abs/2005.11401>
- 26 [1907.10529] SpanBERT: Improving Pre-training by Representing and Predicting Spans  
<https://arxiv.org/abs/1907.10529>
- 27 28 [2006.03654] DeBERTa: Decoding-enhanced BERT with Disentangled Attention  
<https://arxiv.org/abs/2006.03654>
- 29 30 [2004.05150] Longformer: The Long-Document Transformer  
<https://arxiv.org/abs/2004.05150>
- 31 32 [2101.03961] Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity  
<https://arxiv.org/abs/2101.03961>
- 33 [1511.06434] Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks  
<https://arxiv.org/abs/1511.06434>
- 34 58 [2204.02311] PaLM: Scaling Language Modeling with Pathways  
<https://arxiv.org/abs/2204.02311>

- <sup>35</sup> [2304.02643] Segment Anything  
<https://arxiv.org/abs/2304.02643>
- <sup>36</sup> <sup>37</sup> [1509.06461] Deep Reinforcement Learning with Double Q-learning  
<https://arxiv.org/abs/1509.06461>
- <sup>38</sup> [1511.06581] Dueling Network Architectures for Deep Reinforcement Learning  
<https://arxiv.org/abs/1511.06581>
- <sup>39</sup> [1509.02971] Continuous control with deep reinforcement learning  
<https://arxiv.org/abs/1509.02971>
- <sup>40</sup> [1602.01783] Asynchronous Methods for Deep Reinforcement Learning  
<https://arxiv.org/abs/1602.01783>
- <sup>41</sup> arxiv.org  
<https://arxiv.org/pdf/1707.06347.pdf>
- <sup>42</sup> [1710.02298] Rainbow: Combining Improvements in Deep Reinforcement Learning  
<https://arxiv.org/abs/1710.02298>
- <sup>43</sup> [1801.01290] Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor  
<https://arxiv.org/abs/1801.01290>
- <sup>44</sup> [1802.09477] Addressing Function Approximation Error in Actor-Critic Methods  
<https://arxiv.org/abs/1802.09477>
- <sup>45</sup> [1802.01561] IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures  
<https://arxiv.org/abs/1802.01561>
- <sup>46</sup> [1911.08265] Mastering Atari, Go, Chess and Shogi by Planning with a Learned Model  
<https://arxiv.org/abs/1911.08265>
- <sup>47</sup> <sup>48</sup> [2106.01345] Decision Transformer: Reinforcement Learning via Sequence Modeling  
<https://arxiv.org/abs/2106.01345>
- <sup>49</sup> [2006.04779] Conservative Q-Learning for Offline Reinforcement Learning  
<https://arxiv.org/abs/2006.04779>
- <sup>50</sup> [2011.13456] Score-Based Generative Modeling through Stochastic Differential Equations  
<https://arxiv.org/abs/2011.13456>
- <sup>51</sup> [2010.02502] Denoising Diffusion Implicit Models  
<https://arxiv.org/abs/2010.02502>
- <sup>52</sup> [2207.12598] Classifier-Free Diffusion Guidance  
<https://arxiv.org/abs/2207.12598>
- <sup>53</sup> [2102.09672] Improved Denoising Diffusion Probabilistic Models  
<https://arxiv.org/abs/2102.09672>
- <sup>54</sup> [1406.2661] Generative Adversarial Networks  
<https://arxiv.org/abs/1406.2661>

<sup>55</sup> [1606.03657] InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets

<https://arxiv.org/abs/1606.03657>

<sup>56</sup> <sup>57</sup> [2005.14165] Language Models are Few-Shot Learners

<https://arxiv.org/abs/2005.14165>

<sup>59</sup> [2302.13971] LLaMA: Open and Efficient Foundation Language Models

<https://arxiv.org/abs/2302.13971>

<sup>60</sup> <sup>61</sup> [2203.15556] Training Compute-Optimal Large Language Models

<https://arxiv.org/abs/2203.15556>

<sup>62</sup> arxiv.org

<https://arxiv.org/pdf/2205.05131.pdf>