Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

# Temporally stable monocular 3D human pose estimation for tunable body language generation

Áron Hoffmann
[redacted]
hoffmaar@tcd.ie

Supervisor:
Prof. Ivana Dusparic

8th May 2020

A Masters Thesis submitted in partial fulfilment
of the requirements for the degree of
MAI (Electronic and Computer Engineering)

# Declaration

I hereby declare that this project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed: 

Date: 08 May 2020

# Abstract

Realistic body language generation for social robots and animated characters is an extremely challenging problem that has only been attacked recently. The source of the difficulty is both finding suitable data and a good learning method. In this study, we propose a self-supervised, adversarial architecture for learning to generate body language from the text and the voice of the speaker. In contrast with previous attempts, our method takes into account a representation of the personality of the speaker as well. It trains of massive amounts of video available online. We lay out, how this design can be evaluated, quantifying both the realism of the pose and the appropriateness of the character.

We also propose a technique, that can be used to infer some personality traits of the speaker in videos, such as seriousness. This permits the tracking of these personality traits along the duration of speech, paving the way for building reactive agents.

We recognise that the current 3D human pose estimation tools are not accurate enough to support the above project. To remedy this, we lay out a method to train a more robust and stable pose estimator. Stability and robustness are achieved through the utilisation of heavy data augmentation and a combination of best practices from other state-of-the-art architectures. We lay out procedures to evaluate the resulting model, both comparing it to other methods and measuring its stability and robustness. We detail work that remains to be done to create a tool that fulfils our design requirements and enables the learning of body language.

# Acknowledgements

I would like to thank my research supervisor, Prof. Ivana Dusparic for guiding and helping me throughout the process.

To my co-supervisor, Prof. Conor McGinn, for his advice and expertise.

To my peers for their support. Particularly, I would like to thank Rob and Alec for their friendship and support.

Finally, thank you to my mother, father, and sister for the advice, encouragement, and assistance throughout my studies, but this past year in particular.

# Contents

# List of Figures

# Nomenclature

| | | |
|---|---|---|
| $a$ | A scalar | |
| $\boldsymbol{a}$ | A vector | |
| $\boldsymbol{A}$ | A matrix or a higher-order tensor | |
| $a_i$ | $i^{\text{th}}$ element of vector $\boldsymbol{a}$ | |
| $\boldsymbol{A}_{:,i}$ | $i^{\text{th}}$ column of matrix $\boldsymbol{A}$ | |
| $A_{i,j}$ | Element in the $i^{\text{th}}$ row and $j^{\text{th}}$ column of matrix $\boldsymbol{A}$ | |
| $\hat{p}$ | Estimate of p | |
| $\|\boldsymbol{z}\|_1$ | $L_1$ norm of vector $\boldsymbol{z}$, $\sum_i \text{abs}(z_i)$ | |
| $\|\boldsymbol{z}\|_2$ | $L_2$ norm of vector $\boldsymbol{z}$, $\sqrt{\sum_i z_i^2}$ | |
| $\|\text{things}\|$ | Size of set 'things' | |
| $\mathbf{1}_{\text{condition}}$ | 1 if condition is true, 0 otherwise | |
| $\mathbb{E}[\cdot]$ | Expectation | |
| AUC | Area under the curve | |
| AP | Average precision | |
| BN | Batch normalisation | |
| CGAN | Conditional GAN | |
| CNN | Convolutional neural network | |
| DNN | Deep neural network | |
| FFT | Fast Fourier transform | |
| GAN | Generative adversarial network | |
| GRU | Gated recurrent unit | |
| LSTM | Long short-term memory unit | |
| MPJAE | Mean per-joint angle error | deg |
| MPJLE | Mean per-joint localisation error | % |
| MPJPE | Mean per-joint position error | mm |
| MSE | Mean squared error | |
| NN | Neural network | |
| OKS | Object keypoint similarity | % |

| | | |
|---|---|---|
| PAF | Part affinity field | |
| PCK | Percentage of correct keypoints | % |
| PCP | Percentage of correct parts | % |
| RNN | Recurrent neural network | |
| SGD | Stochastic gradient descent | |
| WGAN | Wasserstein GAN | |
| WCGAN | Wasserstein conditional GAN | |

# 1 Introduction

Designing interactive, adaptive, and human-like robots is still more in the realm of science fiction, rather than of research and development. One of the main reasons for this is the lack of necessary datasets. It is even unclear, how such data *could* be collected, for example for emotional and character shifts during speeches and interviews. This is one of the issues we encountered, while trying to design a way to teach Stevie [60], the social robot, some social skills.

Instead, we targeted a somewhat simpler problem of generating body language that goes along with a speech, i.e. the voice and the transcript. Solving this problem has been attempted before. Although not exclusively [35], the general approach for this kind of problem has been using deep learning with a combination of supervised learning, minimising the deviation from the estimated and the ground truth pose [25, 50, 124], and self-supervised, adversarial learning [22, 25]. These approaches do not consider an important aspect of the speech: the person speaking, or their specific body language 'dialect'. Even when it is acknowledged, [25], it is not part of the model, but a separate model is trained for each person. In models with GAN-based adversarial learning [29], the necessary entropy that permits the architecture to learn probability *distributions* is also not provided.

We propose a model, that takes an implicit measure of personality, termed *latent personality* vector. We design a GAN-based training procedure that learns to associate gestures with the latent personality as well as the voice and the semantics of the speech. It can be trained on a massive dataset of TED and TEDx talk videos easily available. This when run, this model generates *samples* from a distribution of gestures, parameterised by the latent personality and conditioned on the audio and the text.

We recognise that this provides an opportunity towards our initial goal, making Stevie adaptive. This is the inverse of the problem that the generator solves (predicting latent personality based on gestures vs. predicting gestures based on latent personality). By training a good gesture generator on a vast dataset and looking at videos of speeches, gradient descent optimisation can be used to find the *instantaneous* latent personality most likely to perform the gestures found on the video. Observing variations in personality

and adaptations in emotions would represent a first step towards creating datasets for learning social skills, as mentioned earlier.

The tasks outlined above proved to be infeasible with the current state of the art, however. Gesture learning needs highly accurate 3D poses[1]. These are estimated from videos using human pose estimators. We evaluated several pre-trained tools for 3D human pose estimation and all of them were found unsuitable. Some estimators confuse joints of the different persons, or of the same person. Some are unusably slow. The biggest issue was, however, 'jitter'. A random error of a few centimeters is included in the estimate of each joint. With the joints combined into a skeleton, this jitter gives a strong appearance of shaking, rendering gestures less recognisable[2]. This is because most estimators work on a frame-by-frame basis and do not take multiple frames into account. Most estimators are also unable to give estimates for scenes, where only the bust of the person is visible. They are research tools designed to perform well on test datasets, not 'in the wild', and are not able to handle occlusion [3]. Some also do not generalise well to scenes, lighting conditions, and clothing styles that are different from their training conditions.

Human pose estimation initially relied on traditional computer vision techniques, Bayesian learning, and constraints such as physically realisable poses [4, 21, 55, 92, 95]. In the past decade, however, it has become almost completely dominated by deep learning, where constraints and distributions are learnt implicitly from data [6, 12, 20, 82, 103, 104, 119, 120]. Beyond the sub-project outlined above, human pose estimation has many important applications, such as autonomous vehicles, action recognition, virtual and augmented reality, robotics, and cinematic animation.

Several methods have been recently published that have a strong potential to improve precision and robustness. However, they are not built together into a single, useful tool. We set out to build such at tool. This tool should be:

- Accurate
- Temporally stable, without jitter
- Robust to non-ideal scenes, rooms, lighting conditions, and occlusion

This tool should be evaluated, both comparing it against other methods and verifying that the above design requirements are met.

[1] 2D pose estimates would be explicitly less useful, as they cannot be mapped to a robot like Stevie or a 3D animated model.

[2] Filtering is not useful in this application, as that would also filter out *intentional* episodes of high velocity and acceleration, making any generated gesture appear overwhelmingly 'stiff'.

[3] This is explicitly specified in some projects, like [126].

This thesis is structured as follows. chapter 2 gives a brief background on deep learning methods, pose estimation datasets, metrics, and a survey of methods for 2D and 3D human pose estimation. It also details the previous and current attempts for generating human body poses. Some details about gradient based network explanation is also given, which is relevant for the human pose understanding sub-project. chapter 3 details the design for the body pose generator architecture, explains the body pose understanding experiment, and presents the proposed 3D human pose estimator. It also contains discussion about some of the design choices made and limitations. chapter 4 describes the work that has been completed so far and gives procedures for evaluating all three sub-projects. Finally, chapter 5 summarises the contributions of the work and lays out future directions.

# 2 Background

Human pose estimation is a field of computer vision that is concerned with locating body parts of a human in an image or a video. This is generally achieved by estimating the location of certain joints or body parts, then combining these to form a skeleton-like model. This is a challenging task and is normally approached by learning from large datasets using machine learning techniques.

Traditionally, human pose estimation has a wide range of applications, such as person tracking, action recognition, VR/AR, animation, or sports analysis. The first project of human pose generation adds to the list of areas, where human pose estimation is useful.

## 2.1 Deep Learning

Deep learning is a machine learning technique that has well-established roots [90, 91], but garnered a significant interest in the last decade due in part to the availability of sufficient hardware [49], in part to the availability of enormous amounts of training data, and in part the astonishing results deep learning based methods have been able to produce [44, 85, 97]. Convolutional Neural Networks (CNNs) [33, 49, 54, 100] refer to a subfamily of deep learning based models that are particularly successful in computer vision tasks. The majority of successful methods for human pose estimation in the last five years have been based on CNNs, as discussed in sections 2.4 and 2.5. Therefore, I will use this section to focus on DNNs and CNNs and provide a brief operational summary. This section omits a lot of the details and arguments inherent to deep learning. For further details, please consult *Deep learning* [28] by Goodfellow, Bengio and Courville.

### 2.1.1 Structure of neural networks

In their most essential form, deep learning models called deep neural networks (DNNs) consist of a sequence of layers. These layers are able to transform the input in complex non-linear ways, but operate on deceivingly simple principles: they are made up of affine

transformations and non-linear *activation functions*.

$$\boldsymbol{x}^{(l)} = f^{(l)}\left(\boldsymbol{W}^{(l)\top}\boldsymbol{x}^{(l-1)} + \boldsymbol{b}^{(l)}\right) \tag{2.1}$$

Here, $\boldsymbol{x}^{(l)}$ represents the output of layer $l$, also called its activation. The output of an $L$ layers deep DNN, $\hat{\boldsymbol{y}}$, is $\boldsymbol{x}^{(L)}$. $\boldsymbol{x}^{(0)}$ denotes the input to the model. $\boldsymbol{W}^{(l)}$ and $\boldsymbol{b}^{(l)}$ are the trainable weights and biases, respectively, corresponding to layer $l$. $f^{(l)}$ symbolises the activation function.

Although historically sigmoid $\sigma$ and the related tanh were used as activation functions [54], rectified linear units (ReLUs) and its various generalisations [34, 62] have been shown to perform better [75] and are the preferred activation function in modern deep learning. See equations (2.2) and (2.4) for reference. All of these functions are applied to the vector $\boldsymbol{x}^{(l)}$ element-wise.

The last layer of a deep neural network is treated differently, compared to the others. When the network is trained to predict binary yes-or-no choices, the activation function of choice is usually the sigmoid function. For categorical selection, softmax (equation (2.3)) is used. For regression problems (estimating the value of one or more real numbers), no activation function is applied in the last layer.

$$\sigma(z) = \frac{1}{1+e^{-z}} \qquad \tanh(z) = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}} = 2\,\sigma(2z) - 1 \tag{2.2}$$

$$\operatorname{softmax}(\boldsymbol{z}) = \frac{e^{\boldsymbol{z}}}{\sum_i e^{z_i}} \tag{2.3}$$

$$\operatorname{ReLU}(z) = \begin{cases} 0 & z < 0 \\ z & z \geq 0 \end{cases} \tag{2.4}$$

The network described above is often called a simple *feed-forward* neural network.

### 2.1.2 Training neural networks

Before the network can be trained, a *loss function* must be selected first, to determine a scalar measure of the error or *loss* $\mathcal{L}$ in the model. For real number outputs, $L_2^2$ error (also known as mean squared error, MSE) or $L_1$ error are common choices. For binary (sigmoid) or categorical (softmax) outputs, cross entropy CE (or the related KL divergence) are used. Various further regularisation terms, such as *weight decay* $\sum_l \left\|W^{(l)}\right\|_2^2$, are also

often added to the loss with appropriate weighting.

$$L_2^2(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \mathrm{MSE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \|\hat{\boldsymbol{y}} - \boldsymbol{y}\|_2^2 = \sum\nolimits_i (\hat{y}_i - y_i)^2 \tag{2.5}$$

$$L_1(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \|\hat{\boldsymbol{y}} - \boldsymbol{y}\|_1 = \sum\nolimits_i |\hat{y}_i - y_i| \tag{2.6}$$

$$\mathrm{CE}(\hat{\boldsymbol{y}}, \boldsymbol{y}) = -\sum\nolimits_i y_i \log(\hat{y}_i) \tag{2.7}$$

The model is trained using stochastic gradient descent (SGD). Gradients $\frac{\partial \mathcal{L}}{\partial W^{(l)}}$ and $\frac{\partial \mathcal{L}}{\partial b^{(l)}}$ are computed through the repeated application of the chain rule of calculus, and weights and biases are adjusted in small steps in the direction opposite to the gradient. If $\mathbb{W}$ represent all weights, biases, and other model parameters and $\eta$ is the learning rate, the update equation of the neural network is

$$\mathbb{W} := \mathbb{W} - \eta \nabla_{\mathbb{W}} \mathcal{L}.$$

Stochastic gradient descent is called stochastic, because the loss $\mathcal{L}$ is not evaluated on the entire batch of available training data, but a random sub-sample called a *mini-batch* that is re-sampled for each update. Various tweaks on SGD have been proposed [45, 108], but the principle remains the same.

Layers such as Dropout [101] or Batch Normalisation (batchnorm, BN) [39] can regulate, normalise, speed up training, and facilitate the propagation of gradients.

Popular tools for implementing the methods detailed in the last two sections include TensorFlow [64] and Pytorch [81]. They contain the necessary abstractions and infrastructure to relatively easily define, train, and use neural networks with good performance. At their core is an engine, which performs the differentiation necessary for SGD automatically.

### 2.1.3 Elements of Convolutional Neural Networks

CNNs are specialised DNNs designed to process images efficiently, with each operation only considering a small patch of pixels. Convolutions have been a staple of image processing even before deep learning.

In a CNN, the matrix multiplication from equation (2.1) is replaced by a **2D convolution operation**. See equations (2.8) and (2.9). The spatial size of $\boldsymbol{W}^{(l)}$, $K_0^{(l)}$ and $K_1^{(l)}$ are

called the *kernel size* of the convolution.

$$\boldsymbol{X}^{(l)} = f^{(l)}\left(\boldsymbol{b}^{(l)} + \boldsymbol{W}^{(l)} * \boldsymbol{X}^{(l-1)}\right) \tag{2.8}$$

$$x_{c,\,s,\,t}^{(l)} = f^{(l)}\left(b_c^{(l)} + \sum_{i=0}^{K_0-1}\sum_{j=0}^{K_1-1} \boldsymbol{w}_{c,:,i,j}^{(l)} \cdot \boldsymbol{x}_{:,\,s-i+\lfloor K_0/2\rfloor,\,t-j+\lfloor K_1/2\rfloor}^{(l-1)}\right) \tag{2.9}$$

Another common element of CNNs is a **pooling** layer, which downsamples the activations according to certain rules. Most recent networks use $2 \times 2$ max pooling, which returns the maximum value in each 2 pixel $\times$ 2 pixel image block and simultaneously halves the activation image size. In some networks, pooling is not used and image size is reduced by applying a *stride* to the convolution directly. In a convolution with a stride of 2, only every second pixel is computed.

Frequently used in pose estimation is the opposite of pooling and strides, the **upsampling** operation, which increases the spatial size of an activation. A common strategy for upsampling is *nearest neighbours*, which duplicates pixels appropriately.

Activation size can also be increased through a **transposed convolution** (sometimes incorrectly called deconvolution), denoted $*^T$, which performs a convolution with fractional strides. A more efficient way to obtain the same result as with transposed convolution has been proposed recently, called **sub-pixel convolution**. [94]. All three of these operations are used extensively in human pose estimation, as discussed in sections 2.4 and 2.5.

In some models, *dilation* is used in convolution layers. This spreads out the convolution kernel wider, leaving a some gaps between the pixels to to which the convolution is applied. This widens the field-of-view of each convolution, without affecting the amount of computation required for it.

Importantly, all operations described above are differentiable, meaning that SGD can still be used to train CNN models. A quick demonstration of common convolution methods can be found at [18][1].

### 2.1.4 Transfer learning in computer vision

The availability of the large-scale ImageNet dataset [15] for image classification has resulted in the creation of high quality *feature extractor* CNNs. These networks are trained to perform well on image classification. If the last few layers are removed, however, they output useful general abstractions or *features* about the image. Therefore, the last few layers can be replaced with layers adapted to object detection or human pose estimation,

[1] `https://github.com/vdumoulin/conv_arithmetic`

for example, rather than image classification. Good results can be achieved on these alternative tasks with little fine-tuning of the feature extractor weights and without expensive and time consuming architecture search. The learning progress made in image classifications can be transferred to other tasks. Although more efficient models have been proposed since [107], many human pose estimation models make use of two feature extractor families in particular: VGG [100] and ResNet [33].

VGG is made up of a sequence of 'thick' convolutions with many input and output channels. ResNet, on the other hand, goes much 'deeper' and is composed of a long sequence of residual blocks. Each residual block made up of 2 or 3 convolutions. The output of a block is equal to the output of the last convolution, added to the block's input. This substructure has proved beneficial to training CNNs, and also used in as part of networks other than feature extractors [76].

### 2.1.5 Data augmentation

Data augmentation is a simple method used to improve the generalisation performance of DNNs and CNNs, without the need to collect more data. It transforms the existing training data in ways that the machine learning model is supposed to be invariant to. Examples of data augmenting transforms for image processing include random crop, random scale, random rotation of the input image, or occluding parts of the image with black or white shapes. Training the model on this extended dataset makes it more likely to be accurate when a test input image is rotated or occluded differently compared to the training images.

### 2.1.6 Generative adversarial networks

Generative adversarial networks (GANs) [2, 29, 69, 74] is an unsupervised machine learning framework with some game theoretic aspects. It was invented in its current form in 2014, but has been around for much longer [93]. They became popularised partly due to the spectacular results they were able to achieve in image generation [44, 83]. GANs are composed of two components: a generator and a discriminator. The generator receives a large vector filled with random noise as input and produces an output that looks like the training data (e.g. images, poses, etc.). The discriminator receives its input either from the generator or from the training set, and must decide, where it came from (i.e. whether its input real or fake). Since the discriminator is usually fully differentiable, this it can be used as a training signal to the generator. Thus, the generator is trained to fool the discriminator, while the discriminator is simultaneously trained to identify the generator's output better. This way, the generator eventually learns to reproduce *samples* from the hypothetical *training data distribution*, parameterised by the random

and conditional inputs. More formally, the loss function of a GAN is

$$\mathcal{L}_{\text{GAN}}(G,\, D) = -\mathbb{E}_{\boldsymbol{x}}\left[f\left(D\left(\boldsymbol{x} \mid \boldsymbol{y}\right)\right)\right] - \mathbb{E}_{\boldsymbol{z}}\left[f\left(-D\left(G\left(\boldsymbol{z} \mid \boldsymbol{y}\right) \mid \boldsymbol{y}\right)\right)\right],$$

where $G$ and $D$ denote the generator and discriminator networks, respectively. $\boldsymbol{x}$ is a real example from the training set, $\boldsymbol{z}$ is a random input, while $\boldsymbol{y}$ is the optional conditioning input. Commonly used functions for $f$ are $f(t) = \log(\sigma(t))$ [29] or $f(t) = t$ [2] [74].

Although the input of the generator is randomly sampled from a uniform or Gaussian distribution, it implicitly represents a *latent* space to the generator, where the combination of some basis vectors decide what the output looks like. These parameters can generally be interpolated linearly to a useful degree. For example, in a generator that outputs images,

$$\boldsymbol{z}^{(\text{woman with sunglasses})} - \boldsymbol{z}^{(\text{woman without sunglasses})} + \boldsymbol{z}^{(\text{man without sunglasses})} = \boldsymbol{z}^{(\text{man with sunglasses})}.$$

Basis vectors for these interpolations can either be found experimentally, or through one of the various methods devised to estimate them [17, 83, 84, 111, 114]. This interpolation makes GANs practical for generating outputs that may not be part of the training dataset, but follow certain parameters.

### 2.1.7 Recurrent neural networks

Recurrent neural networks (RNNs) are a way to model temporal dependencies in deep learning. In its simplest form, it is a DNN that takes two inputs: the actual outside input for timestep $t$ and its own output from the previous timestep, $t-1$. More formally,

$$\hat{\boldsymbol{y}}^{(t)} = f\left(\boldsymbol{x}^{(t)},\, \hat{\boldsymbol{y}}^{(t-1)}\right),$$

where $f$ is some DNN. Various improvements to this form and to $f$. Most commonly used are LSTM [36] and GRU [13].

## 2.2 Pose estimation datasets

A large amount of data is available for research projects to learn 2D and 3D poses. These datasets contain images or videos, with the position of joints of the human subjects annotated. Some datasets provide data beyond the skeleton, such as depth maps or full body meshes. Several datasets categorise poses into activities beyond. Each dataset has a slightly different approach in collecting the images and the annotations, leading to different strengths and weaknesses. The most important datasets are presented here, without an expectation of completeness.

### 2.2.1 2D

**COCO (Common Objects in Context)** [58] is a large collection of annotated images of diverse objects and settings collected mainly from Flickr. It includes around 200 thousand images, where the 2D pose of human subjects have been hand-annotated. A large proportion of these images show several humans, sometimes interacting with each other. The amount and quality of the data make it a cornerstone of object detection and 2D human pose estimation. The images provided are standalone and do not form sequences, thus it cannot be used to learn dynamics. However, it can provide models with useful clues about variation in texture and context.

The **MPII** human pose dataset [1] is somewhat smaller, containing around 25 thousand hand annotated images. Many of these images contain several human subjects. The focus of this dataset is to categorise the poses into one of over 400 common actions and activities. As such, the subjects, clothing, and environment are all diverse. Although the images were extracted from YouTube videos, only standalone frames were annotated, not sequences. Neighbouring frames are provided for convenience without annotation.

### 2.2.2 3D

**Human3.6M** [9, 40] is the *de facto* standard dataset for training, evaluating, and comparing 3D human pose estimators. It is the largest publicly available dataset with high definition video of 11 professional actors and actresses performing 17 types of actions. It is composed of 3.6 million frames, including the test set (which has pose data withheld). The video comes from 4 cameras stationed at the floor, recording simultaneously. The accurate 3D position of 24 joints is recorded by a motion capture (mo-cap) system. Additionally, it also comes with detailed body scans and low resolution time-of-flight (TOF) sensor data. While the dataset provides a large amount of data, it also comes with disadvantages. All of the footage was recorded in the same room, making models trained solely on this dataset prone to overfitting. The performers were wearing mo-cap markers visible on the regular camera input. They also did not perform in varied clothing, likely further limiting the generalisation ability of the models trained on this dataset.

**HumanEva-I** [96] is an older and smaller dataset that shares similarities with Human3.6M. It also uses a motion capture system for the ground truth pose data, but is made up of approximately 40 thousand frames only. It has 4 actors performing 6 types of actions, recorded by 3 colour and 4 grey-scale cameras in a relatively low resolution. Beyond the smaller size of this dataset, its drawbacks follow those of Human3.6M: it is limited by the low variation in location and clothing.

**Total Capture** [110] is another motion capture based dataset with almost 1.9 million

frames from 8 cameras filming 5 subjects. Aside from providing synchronised inertial measurement data for each joint, it shares characteristics with Human3.6M and HumanEva-I.

The authors of **MPII-3DHP** [67] have noted the limitations of the previously available datasets and created a new dataset attempting to alleviate some. This dataset is recorded in a green-screen room, with the aim of facilitating data augmentation and overlaying the subjects upon varied environments. Some of the clothing worn by the subjects were also plain coloured to facilitate easy overlaying of textures for data augmentation. MPII used a motion capture system that does not require the presence of markers on the body, preventing the model from making use of the location of the maker as a hint for the output. The marker-less motion capture system also permitted looser, more varied apparel. The dataset includes 8 actors and actresses performing 8 types of actions. The dataset provides over 1.3 million high resolution frames from 15 cameras at varied angles. This dataset is an overall improvement which augments Human3.6M well, with useful attempts to enable data augmentation techniques. The increasing the number of performers and the variation in real apparel and scenery is still desirable, however.

The **Total Motion** dataset [120] aims to fill some of the gap left by Human3.6M. It was recorded in the CMU Panoptic Studio [42] using 31 hardware-synchronised high definition cameras, looking at highly varied angles. 40 subjects are included, wearing natural clothing. Their body pose is triangulated to a high accuracy from the aggregation of the video feeds. This dataset also includes triangulated annotations for the hands and fingers of the subjects. Importantly, this dataset includes an annotation for each keypoint whether the keypoint is visible or self-occluded in a frame, allowing for algorithms that prefer discarding these joints from training. Altogether, it contains around 834 thousand frames of body annotation. While this dataset does not completely fix the issue of environmental diversity, it ameliorates lack of the diversity of performers and clothing.

**Unite the People (UP-3D)** [53] is based on the images of two smaller 2D pose datasets (the Leeds Sports Pose dataset [41] and the FashionPose dataset [14]), which show subjects in varied environments, often outdoors, wearing various apparel. It uses an automated model to generate plausible 3D poses, which human annotators sort as good or bad fits. Thus, it improves the exposure to environments, subjects, and clothing of any model trained on this dataset. In exchange it sacrifices the accuracy provided by motion capture systems. Furthermore, it provides relatively little data, containing 3D pose (along with body-part segmentation and detailed 2D pose) for just over 8 thousand images. These images do not form continuous sub-sequences (videos), meaning that this dataset has little use in learning temporal stability.

**3D Poses in the Wild (3DPW)** [63] took a distinctively novel approach for enriching

available 3D human pose data. It uses cameras and inertial measurement units (IMUs) integrated into smartphones in conjunction with wearable IMUs at the joints of the subjects. 3D human and camera pose is estimated by a graph-based optimisation model. This approach permits the recording of human 3D pose along with video in natural, outdoor environments ('*in-the-wild*'), where a motion capture system is not present. Clothing worn by the subjects is also diverse. The dataset contains over 50 thousand frames. Under the ideal conditions of the Total Capture dataset [110], the method used for estimating the 3D poses reaches a mean joint error of only 26 mm. However, the demo video on the project page[2] shows pervasive errors like missed frames, noisy joints, or varying scale. While the direction is promising, more work is needed before this dataset can be relied upon for accuracy.

**SURREAL** [112] is a large synthetic dataset. It is made up of 6 million frames of CGI (computer generated imagery) based humans inserted into model environments and rendered in a photo-realistic manner. The poses of the humans were mapped based on motion capture data. As the entire environment is synthetic, a large amount of data can be collected, such as depth, optical flow, or even surface normals. However, CGI based images lack some of the complexity and noise of real photos and videos, and some CNN-based models may struggle to generalise between real-world and computer-generated textures.

## 2.3 Pose metrics

When evaluating pose estimates, it is crucial to select reliable metrics to quantify their accuracy. In the following, the metrics most frequently employed by the literature are summarised.

**MPJPE (mean per-joint position error)** is the benchmark metric for several 3D pose datasets, such as Human3.6M and HumanEva-I [40, 96]. It is simply the Euclidean distance between the estimated and the target positions, averaged over all joints: see equation (2.10). A lower MPJPE means the estimated joint positions are closer to the target joints, indicating a better model. This metric is said to be less robust [40], as it can be overwhelmed by both a single, badly predicted joint and many well-predicted joints with minor errors. The related **PA-MPJPE** is also in use. In this metric, a rigid-body transform called *Procrustes Analysis* is used to shift, rotate, and scale the estimated skeleton before computing the MPJPE. This metric is designed to penalise the errors in the pose itself, rather than its alignment with the coordinate system. A further related metric is **MPJAE (mean per-joint angle error)**, where the mean is computed over the angles between neighbouring joints. As most contemporary methods predict joint

[2] `https://virtualhumans.mpi-inf.mpg.de/3DPW/`

positions, rather than joint angles, this metric is seen applied less often.

$$\text{MPJPE}\left(\hat{p}, p\right) = \frac{\sum_{i \in \text{joints}} \left\|\hat{p} - p\right\|_2}{|\text{joints}|} \tag{2.10}$$

**PCP@$\alpha$ (percentage of correct (body)parts)**, described in [19], is the fraction of body parts estimated correctly. It counts a body part correct if its estimated length (the distance between its two end joints) is within a threshold fraction $\alpha$ of the expected length for that body part: $\frac{|\hat{l}_i - l_i|}{l_i} < \alpha$. A common value for $\alpha$ is 0.5. The /a larger value of PCP indicates a better model. This metric gives small limbs much harsher error margins than large limbs, is relatively forgiving about joint position. It can be most often seen used in 2D pose estimation.

**PCK@$\alpha$ or PCKh@$\alpha$ (percentage of correct keypoints)**, as used in [109], is the fraction of joints (i.e. keypoints), where the estimate is within a threshold distance of the target. For PCK@$\alpha$, this threshold is taken as $\alpha$ times the diameter of the torso. For PCKh@$\alpha$, the threshold is $\alpha$ times the lenght of the 'head bone', the distance between the bottom of the neck and the head joints. For PCK, particularly in 3D estimation, the arbitrary threshold of 150 mm is also used sometimes [67]. This metric can be further extended and made more informative computing it over a range of thresholds and integrating it to from the **AUC (area under curve)** or **AP (average precision)**. While PCK was initially a 2D metric, it is now also in use in 3D estimation. Some authors advocate for its use in 3D [67], claiming that it is more robust, and preferring it over MPJPE. [40] proposes this metric for 3D under the name of **MPJLE (mean per-joint localisation error)**. A higher PCK corresponds to a better model.

**OKS (object keypoint similarity)**, proposed in [88], is a novel metric, targeting 2D estimation in particular. It interprets the joint estimates as realisations of a 2D Gaussian distribution over the image. The mean of this distribution is the ground truth location for that particular joint. The variance is the variance amongst human annotators for the joint ($k_i$ in equation (2.11)), scaled based on the size of the skeleton within the image ($s$ in equation (2.11)). The OKS is equal to the mean of the unnormalised probability density value sampled at the estimated location for each visible joint, as shown in equation (2.11). The goal of this metric is to increase robustness. It assigns little error to differences that human annotators cannot easily differentiate, like the precise pixel-wise position of the estimate, while it assigns a large error to blunders where the estimate no longer corresponds to the correct joint. OKS ranges from 0 to 1, with a higher value indicating a better prediction.

$$\text{OKS}\left(\hat{p}, p\right) = \frac{\sum_{i \in \text{joints}} \exp\left(\frac{\left\|\hat{p}_i - p_i\right\|_2^2}{2 s^2 k_i^2}\right) \mathbf{1}_{\text{joint } i \text{ visible}}}{\sum_{i \in \text{joints}} \mathbf{1}_{\text{joint } i \text{ visible}}} \tag{2.11}$$

## 2.4 2D pose estimation

To solve 3D pose estimation, the simpler subtask of 2D pose estimation must be solved first.

### 2.4.1 Common design choices

**Deep learning:** The vast majority of well-performing recent techniques use the deep learning framework. Some outdated approaches [4, 55, 92, 95] use Bayesian methods, that manipulate explicit probability distributions to satisfy constraints. These approaches are also mostly concerned with 3D pose estimation.

**Top-down vs. bottom-up detection:** Many images, where human pose must be estimated, do not only contain a single human, but several. Two approaches are generally in use to go around this issue. The more straightforward one is the top-down approach, where object detection tools are first used to identify the locations of all humans in the image, which are then cropped. Top-down pose estimation algorithms simplify multi-person pose estimation to estimating the pose of the single person at the centre of the cropped regions. Most techniques discussed in section 2.4.3 are either single-person or follow a top-down approach.

However, humans often overlap or interact, making it impossible to cleanly crop images to individuals. The bottom-up approach [6, 7, 48], instead, pinpoints all skeleton parts (body parts or joints) on the image, then uses matching algorithms to find the best combination. As a downside, however, these matching algorithms are usually computationally expensive and non-differentiable.

**Repeated refinement:** A recurring pattern in pose estimation architectures is defining a large stage and using it repeatedly to build more accurate estimates [6, 7, 76, 109, 116]. These stages are chained sequentially, the output of one stage becoming the input of the next. Some papers use a technique called intermediate supervision, where the same learning target and loss function is applied to all stages.

**Multi-scale processing:** It is clear that processing the image at several scales is necessary for accurate human pose estimation. For example, the position and orientation of the arms and hands provide useful clues to for the location of the shoulder. A located shoulder, in turn, help locating the arms and hands better. Some authors [76, 102] try to aid this process explicitly by building it into the network architecture directly.

**Output types:** Human pose estimation is generally considered a regression problem [104, 109], meaning that the target is an estimate of a set of real numbers. Yet, the outputs of the neural networks include some variation. DeepPose [109] used this directly, fully-connected layers to output real numbers directly from the image. Most approaches, however, output a heatmap for each joint. A heatmap is a single channel image, with the value of each pixel representing the unnormalised likelihood of the joint being located at that pixel. It is generally assumed, that the generated heatmaps resemble Gaussian distributions, so location estimate for each joint is then taken as the location with the maximum value (argmax, mode) in the respective joint's heatmap. More recently, Integral Pose Regression [104] argues against this approach and instead uses the expectation (soft-argmax) of the heatmap for the pose estimate. Vector fields (represented as sets of 2-channel images) are also a useful network output to refine estimates and learn relationships between parts [6, 48].

### 2.4.2 Tools

Many, though not all researchers release their research code and some trained model as open source following the publication of a study or paper (e.g. [76, 102, 116]). However, they ten are often not maintained, some are convoluted to account for different research scenarios and difficult to run due to dated dependencies. Their purpose is research and verification, not continued use.

Fortunately, there are also maintained libraries to contrast. These are written to be easy to use and well documented. They are maintained, keeping track of changing libraries and environments, so that running them is as easy as downloading (and in some cases, compiling). Their purpose is to be used as reliable tools.

Most popular of these tools is OpenPose[3] [6, 7, 98], detailed in section 2.4.3 on page 17. It runs on a wide variety of input formats, operating systems (Linux, Mac, Windows), and platforms (CPU, GPU, Nvidia Jetson, Unity). It uses a bottom-up architecture, and can scale to various hardware budgets. It is maintained with regular bug fixes and occasional feature additions. It is also trained for hand, finger, and feet pose estimation.

One of its main 'rivals' is AlphaPose[4] [20, 56, 122], which uses a completely different, top-down approach, based on a refined object detection algorithm detailed in [20]. It claims better precision than OpenPose and is constantly optimised for even better speed and results.

Another easy-to-use and well-maintained tool is Detectron2[5] [119]. It is a framework

[3] `https://github.com/CMU-Perceptual-Computing-Lab/openpose`
[4] `https://github.com/MVIG-SJTU/AlphaPose`
[5] `https://github.com/facebookresearch/detectron2`

mainly based on Mask R-CNN [32], a landmark architecture for object detection and semantic segmentation. Beyond keypoint detection, it has several other useful 'heads' that can be attached, like object detection, instance segmentation, or dense pose estimation.

### 2.4.3 Some notable approaches

**DeepPose** [109] is the first attempt at using CNNs and deep learning to perform pose estimation from 2014. Toshev and Szegedy interpret the setting as a simple regression problem and solve it using 5 convolutional layers followed by 2 fully connected layers. It is a top-down approach and regression outputs are normalised to the person bounding box. The project demonstrates the feasibility of using deep learning for pose estimation and its ability generalise across datasets. The accuracy (PCP) it achieves counted as state-of-the-art at the time, but the skeletons it produces only look approximate. In order to improve the accuracy of the estimates, they also use a multi-stage approach, where all stages are based on the same architecture, but are trained to estimate the error of the previous stage. The authors found that using a second stage adds a few percentages to the accuracy, but the effects are diminished for more than two stages.

While DeepPose treats the pose as a direct regression problem, trying to estimate the joint coordinates as real numbers, almost all techniques in the following take a different approach. Instead they predict *heatmaps* for each joint. A heatmap is a 2D single-channel image, with each pixel representing the unnormalised likelihood of the joint being located at that pixel. The location estimate for each joint is then taken as the location with the maximum value (mode) in the respective joint's heatmap.

Many network architectures following DeepPose, including the notable example of **Convolutional Pose Machine** [116], build on the idea of stage-wise refining the pose estimate. They use a sequence of large stages with repeated structure, but different weights. The same target and loss function is applied after each stage (a technique named *intermediate supervision*), making each stage rely on the previous stage's estimate to make a better one. In the case of Convolutional Pose Machines, the stages are composed of large $11 \times 11$, $9 \times 9$, $5 \times 5$, and $1 \times 1$ convolutions, along with max pooling. Stages 2 and onward receive both the output of the previous stage, and the input image, with a separate set of convolutions applied.

Convolutional Pose Machines only perform top-down processing. Resolution only ever lowers (through pooling) and low-resolution abstractions cannot influence higher-resolution processing in a later stage. The authors of **Stacked Hourglass** [76] note that interleaved steps of top-down and bottom-up processing is necessary. Top-down processing is achieved through repeated $3 \times 3$ convolutions in the form of residual blocks

[33] combined with max pooling, while the bottom-up processing is simply made of nearest-neighbour image upsampling and further residual skip connections. This *hourglass module* is similar to the U-Net architecture [89], which is successful in the related field of semantic segmentation. Newell et al. argue, that stacked application of this hourglass module and using intermediate supervision helps the network to consolidate features at different scales, leading to a better understanding of the image and estimate of the pose.

More recently, **HRNet** [102] focuses on the combination of top-down and bottom-up processing to improve results and amalgamates them into a joint '*exchange unit*'. The network is made up of several streams, each processing the image using residual blocks [33], working at different resolutions. After every four residual blocks, all streams merge information with one another in an exchange unit. In these units, higher resolution streams pass their activations down through a set of strided convolution layers, while lower resolution streams pass activations up through a set of blocks composed of an upsampling and a convolution. Intermediate supervision is dropped, as it did not bring an improvement of performance on this architecture. At a comparable computational requirement and number of parameters, HRNet outperforms previous approaches by several percentage points and shows that in the OKS and PCKh metrics. It demonstrates that multi-scale processing is a useful and important tool for human pose estimation.

Xiao et al. intend to put a point of reference into the increasingly chaotic landscape of complex network architectures. **Simple Baselines** [121] uses transfer learning to extract features from the image (generally a form of ResNet), followed just by a few layers of transposed convolution to finally arrive at a set of heatmaps. It produces surprisingly precise results, achieving state-of the art in some metrics at the time of its publication. This architecture of feature extraction followed by transposed convolution is used by several other works. These include **AlphaPose** [20, 56, 122] and **Detectron/Detectron2** [27, 119], both of which are built on top of object detection architectures [32, 86].

**OpenPose** [6, 7, 98] is one of the cornerstones of human pose estimation. It is the most popular and widely applicable library for 2D human pose estimation. It uses a bottom-up approach of part confidence maps (heatmaps of body parts) and part affinity fields (PAFs). A part affinity field is a 2D vector field, where the orientation of each vector represents the orientation of the body part under that pixel (or **0** if the pixel is not part of a body part). This PAF is output from the network as an image with 2 channels per body part. The network itself is generally similar to the architecture described in Convolutional Pose Machines [116]. The input image is first processed by part of VGG [100]. Then, the first stage of convolutions estimates the PAF and stages 2-6 estimate and refine the part confidence maps. Following the part confidence maps and the PAFs, OpenPose uses traditional, non-learned algorithms to arrive at a list of skeletons.

The part confidence maps are transformed into body part candidates by non-maximum suppression and bipartite matching [52] matches part candidates to skeletons optimally with the help of the PAFs. While more accurate architectures have been proposed, it is packaged as a simple-to-use tool that runs on a wide variety of platforms.

**PifPaf** [48] furthers OpenPose's approach of vector fields and confidence maps. It is another bottom-up architecture and its network predicts part intensity fields (PIFs) and part association fields (PAFs). Each joint estimated has its own PIF, which is composed of a 2D vector field and two scalar images. At each pixel in the PIF of joint $j$, the vector is trained to point at the nearest joint of type $j$ and the two scalar images estimate the size of that joint on the image and the confidence in these estimates, respectively. The network also estimates one PAF for each body part of the skeleton (connection between joints). A PAF is made up of two 2D vector fields and three scalar images. In the PAF of body part $p$, the vectors in the vector fields point to the two end joints of the nearest part $p$, two of the scalar images estimate the sizes of the respective joints, and the third image estimates the confidence in the predictions of the pixel in question. These fields are then fused into skeletons using a greedy matching algorithm. PifPaf significantly outperformed OpenPose and demonstrated the benefit of explicitly estimating relationships and orientations of joints and body parts.

While most previous works used the mode(s) of the generated heat maps, **Integral Pose Regression** [104] instead normalises the heat map into a probability mass function using the softmax function and computes the expectation of the joint position with respect to that distribution. Thus it uses the *soft-argmax* operation instead of argmax. This has a two-fold advantage. First, it takes more of the abstracted information into account during inference. The heat maps freely generated by the neural networks have been shown to be somewhat dissimilar from the Gaussian distribution [38, 61, 78], and the exact pixel selected by argmax introduces a substantial amount of noise. Instead, the expectation over a distribution is influenced by the entire distribution and is less prone to error. Second, the argmax operation is not meaningfully differentiable and heatmap targets must be generated for training. The soft-argmax operation is differentiable, so end-to-end training may be used. Empirically, soft-argmax leads to a better pose estimate for single-person estimation. However, it cannot trivially deal with a multi-mode distribution, such as the one arising in multi-person pose estimation.

## 2.5 3D pose estimation

Beyond simply pinpointing the location of joints on the image, 3D pose estimation also estimates a depth component for each joint, providing a full skeleton that can potentially be used in 3D computer graphics or mapped onto the joints of a humanoid robot. This

is a significantly more challenging task than 2D pose estimation.

### 2.5.1 Some notable approaches

**How much harder is 3D than 2D?** Martinez et al. investigated in [65], whether it is much more difficult to estimate a 3D pose than a 2D pose. They used a decidedly simple model to extract 3D poses from 2D pose estimates only. The results of this model improved by over 25 % when supplied with the ground truth 2D positions instead of positions estimated by a Stacked Hourglass model (discussed in section 2.4.3 on page 16). This places the remaining nearly 75 % of the MPJPE error, 45.5 mm, on the estimation of joint depth. Interestingly, when using 2D ground truth poses as input, the estimations are remarkably smooth and noiseless, even without taking time into account as part of the model. This implies, that if 2D predictions are sufficiently accurate and stable, estimating good 3D positions is also feasible.

**2D pose + matching:** Chen et al. in introduce [11] an interesting way to provide a baseline for 3D human pose estimation. Instead of using a deep learning based model for the depth information, they compile a large library, mapping 2D poses to 3D poses. During inference, they use Convolutional Pose Machines (section 2.4.3 on page 16) to extract 2D coordinates and use them as a key to select a 3D pose from the library. The selection is made based on the Euclidean distance between the respective joints of the 2D pose and the projections of the library (nearest neighbour). While the results are neither particularly accurate, nor smooth, it is one of the simplest possible methods, so it is a basis that more sophisticated method can be compared against. In a sense, it calibrates the error metrics used in 3D pose estimation.

**Using CNNs:** [57] is amongst the first presentations of using a CNN to directly estimate 3D joint locations. They broke the problem down to two tasks: joint location estimation and joint detection. The first outputs a the position of each joint relative to its parent joint as a vector, while the second predicts, whether the joint is present in the cropping frame. Although the results on Human3.6M are inaccurate by today's standards, the method represents a first step and a proof of concept, which can be improved upon in later works.

**Integral Pose Regression** [104] by Sun et al., described in section 2.4.3, can also be easily extended into the depth dimension. Instead of a single heat map per joint, multiple heat maps can be generated with a depth value assigned to each. This collection of heat maps can be interpreted as a 3D probability mass distribution for a joint when normalised using the softmax function. Therefore an expectation of the 3D position can be computed

in a way compatible with back-propagation. The model achieved state-of-the-art results on Human3.6M at the time of its publishing.

**2D marginal heat maps** [77] recognises that 2D pose heatmaps are, in fact, marginal distributions of the full 3D pose heatmaps. Instead of predicting many heatmaps *conditioned* on depth as in [104], Margipose predicts 3 *marginal* heatmaps, for the $xy$, $xz$, and $yz$ planes, respectively. They also use regularisation on the shape of the heatmaps to make them more localised. The authors implement a multi-stage estimator with intermediate supervision and residual blocks, somewhat resembling Stacked Hourglass [76] (section 2.4.3 on page 16), and reach competitive results.

**Monocular Total Capture (MTC)** [120] uses a CNN almost identical to an older version of OpenPose [7] (section 2.4.3 on page 17) as its first step, the difference being that the vectors in the PAFs it generates are 3 dimensional, rather than 2. Furthermore, it does not simply estimate a skeleton, but a parameterised full-body mesh [43]. To achieve this, rather than estimating the pose parameters directly, as most other models do, this model establishes a loss function and performs a form of gradient descent optimisation to arrive at the best pose. This optimisation takes several factors into account:

- The difference between the 2D projection of the 3D pose and the 2D OpenPose estimates
- The alignment of the body parts with their respective PAFs
- Optionally a prior about what poses are likely, learned from the training dataset
- Regularisation terms

MTC performs this optimisation not only for body pose, but for hands, feet, and face as well.

Additionally, when temporal consistency is enabled for videos, a second pass of optimisation is also performed. This pass includes additional terms in the loss function. MTC projects the frame onto the mesh, extracting the texture. Then, it uses this texture to project the next frame's mesh onto the next frame. MTC minimises the optical flow between the two projected frames, along with a smoothness constraint. While the results published are spectacularly accurate and smooth, the repeated optimisation proves to be a very time consuming and computationally expensive procedure.

**Explicit Spatio-temporal Joint Relation Learning:** Sun et al. propose a method in [103] (shortened **ExplicitPose** in the following) to improve accuracy by explicitly targeting not only the locations of joints, but the relationships between them. The create

a CNN very closely based on the Simple Baselines CNN (section 2.4.3 on page 17). Their network has three sets of outputs:

- A heatmap for each joint, localising it on the image
- Relations to other joints. Specifically, a 3D vector field for each joint, estimating the direction and distance to its parent joint. The impact of each vector in the vector field is weighed during inference by the heatmap.
- Relations of a joint to itself in neighbouring frames. A 3D vector field is estimated for each joint and temporal difference combination. Several configurations for frames were tried and the best accuracy were found when estimating displacement for all of the following distances from each frame: -3, -2, -1, 1, 2, and 3. Performance or latency was not taken into account. The impact of each vector in the vector field is, again, weighed during inference by the heatmap.

Just like in MTC, a second round of optimisation is used to arrive at the final estimate. Without the repeated rounds of texture extraction and re-projection, however, this optimisation is much less expensive computationally.

The authors argue that this temporal relation learning is an improvement over optical flow. Instead of trying to make use of a generic optical flow (often itself estimated by a CNN), the network is encouraged to learn the part of the motion that is useful to making accurate pose estimates, and do that more precisely.

Interestingly, the temporal relation learning appears to be robust to variations of frame rate, probably because the network is shown both frames of the relation when making its estimate.

This approach can also be compared with the VideoPose3D method (section 2.5.1). VideoPose3D uses simple 1D convolution passing through the sequence of frames in a video, considering some neighbouring frames at each step. ExplicitPose also passes over neighbouring frames in a manner similar to convolution but it performs a more complicated and non-linear function over them.

**VideoPose3D** [82] is a method for estimating 3D pose in videos based from 2D pose estimates, rather than the video frames directly. The CNN uses 1D dilated convolutions, arranged in residual blocks. The dilation increases the receptive field of each convolution enormously, allowing the model to quickly learn temporal dependencies with several frames difference. Although the accuracy saturates after a few tens of frames of total receptive field, the best performing model used in the paper has a total receptive field of 243. The model also makes use of a form of self-supervision during training, requiring estimated 3D poses to remain faithful to the 2D poses, when projected back. The

method gives strikingly smooth results on videos under ideal conditions. However, it has no ability to handle occlusion.

**Occlusion and geometry awareness:** There are approaches that try to encode prior knowledge about the mechanics of 3D human poses with the aim of improving accuracy. Several methods, such as [82, 127], penalise changes in bone lenght. [87] Tries to learn to predict, what the pose would look like in different camera orientations, thus gaining a better understanding of the 3D pose. [12] uses a cylinder-based person model during training to understand, which joints are self-occluded in each frame. Discarding these unreliable joints for training improves the accuracy of the estimator significantly. The CNN uses Stacked Hourglass for 2D pose estimation and temporal convolution based model similar to VideoPose3D for 3D pose estimation.

### 2.5.2 Multi-person 3D pose estimation

Most 3D pose estimation design proposals are not concerned with multiple persons being in the source photo and expect a scene with a single person. [68] introduces a bottom-up approach to deal with this issue (see section 2.4.1 on page 14 for the top-down vs. bottom-up distinction). They build on OpenPose's approach [6] (section 2.4.3 on page 17) for PAF based estimation and matching. They create a network, which outputs *location maps*: images, which give a location estimate for each pixel. They are able to build a large amount of information redundancy into their algorithm, improving its ability to deal with occlusion and nearby persons.

In contrast, [70] uses top-down processing. They use three separate CNNs: one to detect persons in the image and estimate bounding boxes, one to estimate the distance (root) relative to the camera and the size of the person, and a third for estimating pose within the cropped bounding boxes (based on Integral Pose Regression [104], section 2.5.1 on page 19).

### 2.5.3 In the wild estimation

As accurate 3D pose ground truth is difficult to obtain, most 3D pose estimators are trained on exceedingly homogeneous data: few actors indoors in the same room, performing the same actions, with little variation in apparel. This situation is likely to lead to overfitting, where the estimator performs well on the validation and testing part of the data, but does not perform well on unseen images and videos recorded in different environments, i.e. 'in the wild'. A few attempts have been focused on circumventing this issue.

**Training data** One of the simplest ways to improve the diversity of training data is to estimate the 3D pose based on the 2D pose, rather than the image. This way, a 2D pose estimator model may be used, which is pretrained on diverse 2D pose datasets such as COCO. The simplicity comes with the cost, however, that the image can no longer be taken into account to decipher the depth information. Methods using this approach include [65], [11], and [82].

**Weak/Self-supervision** A more advanced approach is to make use of prior knowledge and weak/self-supervision. It can be generally assumed that the ratio between the lengths of various joints changes little among people. Therefore a term enforcing this assumption can be added to the loss, allowing all output paths of the network to be trained even on images where the depth information is not available. This approach is taken by [82, 127]. It can also be assumed that the 2D projection of the estimated 3D pose should be the same as the 2D pose. Any deviations can similarly be included in the loss, providing extra gradients without requiring more data. Examples of methods including this factor are [115], [3], and [120].

**Adversarial learning** has also been proposed to add supervision losses to models without additional ground truth data. [123] proposes a GAN-based architecture. It has a discriminator model learning to decide whether the pose is estimated or ground-truth. The estimator model is trained to fool the discriminator instead. This architecture results in some accuracy improvement, but is still limited by the lack of diverse ground truth 3D pose data both while training the discriminator and during the validation phase.

### 2.5.4 Temporal consistency

Few past works have studied the exact temporal discrepancy of in pose estimates quantitatively, but experience shows that results often jump and jitter a lot in a random-like motion. This is confirmed by the number of attempts aiming to give more stable estimates [3, 12, 37, 82, 120]. While the state of the art MPJPE error is decreasing with time, the average $\sim 50+$ mm per-joint error is distributed unevenly across video frames unless special care is taken. This is acceptable for some applications, such as activity detection, but seems visually incorrect to the human eye.

Note that the field of human pose tracking [26, 79, 121] generally refers to tracking of persons with the goal of assigning a consistent identity across frames and not tracking of joints with a goal of smoothness.

**Recurrent neural networks** have been investigated with the expectation that they would be able to abstract some of the dynamics of the human pose and yield smooth

results. In particular [37] uses a model traditionally used for machine translation [36, 105]. Although it did gain in MPJPE accuracy, demos show that the results are not much smoother.

**Inference-time optimisation:** Some works try to improve smoothness by establishing an error manifold over the body coordinates, which includes an inter-frame error term. They then perform gradient descent during inference to find the best pose. [3] simply tries to minimise the difference between the pose of two consecutive frames, achieving little gains. MTC [120], on the other hand, performs a complicated texture extraction and re-projection procedure over a full-body mesh, minimising the optical flow between the re-projection and the actual next frame, as summarised in section 2.5.1 on page 20. Little work was done to evaluate or quantify its accuracy following this optimisation, it brings unprecedented accuracy qualitatively [6]. It comes at a cost, however. [3] reports requiring 8 minutes for a 10 second video, or approximately 2 seconds per frame. [120] does not report speed, but took approximately 3 hours per 20 second clip in my experience, or roughly 18 seconds per frame, 2½ orders of magnitude slower than 'real time'.

**Temporal convolution:** Another promising direction of research aiming to stabilise video is the use of temporal convolution, as in VideoPose3D [82] [12] (section 2.5.1 on page 21 and on page 22). This approach computes the output of a 1D CNN over some abstraction from each frame, usually the 2D pose coordinates. The convolution is able to consider multiple frames at the same time and perform a non-linear function over them. It can gain some understanding of the motion of the subject and try to correct for any erroneous motion. This approach is simple, efficient, and effective, yielding very smooth poses on ideal videos. Its performance tends to deteriorate, however, when conditions are not ideal, such as when only a part of the body is visible.

**Explicitly learning temporal relationships** [103] tries to force the CNN to understand the motion better. Requiring to estimate the movement of joints between various frames can regularise the jitter, which cannot be accounted for by the video. In contrast with optical flow based methods, which can fail in scenarios relevant to pose estimation (changing lighting, non-rigid body, uniform texture), this approach is taught to learn the motion of joints of interest *specifically*. Also, like temporal convolution based methods, it is taught to perform a non-linear function over a set of neighbouring frames to improve the estimates. Experiments in [103] show some improvement due to this method.

[6] `https://youtu.be/rZn15BRf77E`

### 2.5.5 Tools

The tools publicly available to perform 3D pose estimation are not nearly as abundant as for 2D. For many promising projects, code and pretrained networks are not made available [12, 103, 104]. Most remaining ones are written with the goal of research and not usage. They are not maintained and are left severely out of date. It takes a considerable amount of time, before they can be installed and their operation can be understood. I detail my experiences with them in chapter 4.

## 2.6 Human pose generation

Human pose generation, or gesture generation, solves a problem similar to human pose estimation. Its output is a sequence of skeleton coordinates given inputs such as text or voice. Unlike human pose estimation, where the output is constrained to strictly correspond to the pose in the input image, human pose generators enjoy a considerable freedom in the output. The target is to be perceived real, not to be numerically accurate. Accordingly, it is somewhat more difficult to train and evaluate.

Human pose generators have various applications, such as conversational agents in robotics, AR/VR, video games or cinematic animation. Due to its complexity, this problem was traditionally attacked by using naive, manually designed, rule based methods. Future machine learning based methods have a potential to be widely useful and achieve remarkable results. There have been a few studies about their realisation very recently.

**Gesture typology:** The study of gestures distinguishes amongst four major types of gestures [51, 66]: beat, deictic, iconic, and metaphoric. Beat gestures are the simplest, emphasising certain syllables and following the rhythm of the speech. Deictic gestures point towards an object or in a certain direction. Iconic gestures represent something being described by the speaker, while metaphoric gestures represent an abstract idea, such as a thumbs up. The timing and the tone of the speech is essential for beat gestures, while semantic information is necessary for the most types. Some works described in the paragraphs distinguish among these gesture types.

**Is it a realistic goal?** Wolfert et al. studied in [118], whether a current machine learning based approach to gesture generation is perceived more natural by human subjects. They used the design from [50] by Kucherenko et al. They make use of a variational autoencoder (VAE) architecture: a non-linear dimensionality reduction tool made up of an encoder and a decoder part. This is trained to abstract a latent representation about pose sequences. Next, the encoder is discarded and is replaced with one trained to produce the latent representation of the corresponding pose for a given voice, thus creating a path

from voice to gestures. The design was trained on a small dataset that includes almost 3 hours timed audio and pose of two individuals. Subjects in the user study watched a video of a skeleton performing gestures generated for a speech audio clip. The machine learning generated gestures were preferred significantly over manually crafted, timed, and random gesture patterns.

**TED from text:** Yoon et al. tried to increase the amount of training data available to them by using TED talk videos [124]. They cited difficulties using previous work to estimate 3D poses from the videos. This mirrors my experience. To circumvent this issue, they discarded 87 % of their data, where conditions are not ideal, and and trained a simple DNN to convert 2D poses from OpenPose [6] to 3D poses. The accuracy of this estimator is not disclosed. The authors also broke up the video into separate scenes using [8]. They published the resulting dataset, including YouTube links, scene limits, 2D, and 3D pose estimates, totalling to 14 000 shots or 53 hours of video.

To learn poses, they used a very simple RNN model that is traditionally used in machine translation tasks [105], augmented by neural attention [5]. The model does not take the voice or the identity of the speaker into account, only the transcript, transformed into simple word embeddings. They simply minimised the mean squared error from the ground truth pose and the velocity of the generated poses, while maximising the variance of the pose sequences. The authors observed the synthesis of all four gesture types. They also mapped the generated poses to a robot prototype. In a perception-based study, the authors found that the proposed method outperformed random gestures in anthropomorphism and speech-gesture-correlation, but had little gain over a baseline method that selected a gesture from a dictionary based on text similarity. Participants were divided, whether the motion corresponded to the speech, likely due to the lack of audio input to the model.

**Personality-based gestures from audio:** Ginosar et al. recognised that personality impacts gestures significantly. They devised a GAN architecture that uses an U-Net-like block [89] with 1D convolutions to generate 2D gestures [25]. Personality is not an input to the model, rather the same architecture is trained on 10 public individuals, such as TV show hosts, separately. Altogether, they used a considerable 144 hours of video for the training and used OpenPose-generated [6] 2D poses as target. The lack of depth information makes it more difficult to potentially remap these poses to a robot. For quantitative evaluation, the authors use the $L1$ distance and the PCK metrics (section 2.3 on page 13). This is somewhat ill-fated, as the goal of the network is to produce plausible gestures, not exact ones. Nevertheless, the architecture was shown to learn significantly different pose sequences for different individuals. In a perception-based study, where they asked participants whether a gesture corresponded to the audio or not, they found that

the generated gestures were on par with gestures selected randomly or simply based on audio similarity.

**Multiple objectives:** Ferstl et al. used [22] used an RNN and GAN-based neural network with several adversarial components to produce motion. Each adversarial component has a different view of the model inputs and outputs and adds different penalties to achieve overall realism. The generator takes the voice as input, tries to explicitly predict the gesture phase at the moment in question, which it then also uses to produce a pose output. To aid their efforts, they recorded a dataset of over 6 hours of a single actor talking and gesticulating. The datasets consists of the actor's voice, his pose captures by a mo-cap system, and partial hand annotations about the phase of gestures he is in at any moment. The approach makes clever use of additional constraints to make better use of limited data, but little evaluation was performed.

**Gesticulator:** In a study contemporary with my work (published after I moved my focus to 3D pose *estimation*), Kucherenko et al. built a gesture generator [51] that bears some similarities to my design. They used two inputs for their network: the text timed and transformed to latent features using BERT [16], and the voice transformed to a mel-spectogram image. The network outputs a pose corresponding to the end of the input. The architecture itself is a relatively simple deep neural network, with some added conditioning on the previous frame's pose. Mean squared error of joint position *and velocity* were used as loss. The network was able to organically (without specific instruction) learn to generate not only beat gestures, but some iconic and metaphoric gestures as well. With quantitative analysis, they found that the gestures generated by all of the proposed model variants were significantly slower than the ground truth gestures. They also performed a perception-based ablation study, observing that both voice and text are essential inputs to their design. They also showed that the PCA dimensionality reduction introduced in [124] is detrimental to the naturalness of the learned gestures. Although videos show that it is clear that the model learned gesture-like motion, the majority of participants ($\sim$ 90 %) preferred the ground-truth pose sequences over the generated ones.

**Normalising flows:** MoGlow [35] is an interesting design that uses a more novel, probabilistic approach to machine learning called normalising flows [46] to synthesise controllable general motion, such as the motion of an animal or a human. It is also contemporary with my work. The architecture of MoGlow is grounded both in invertible transforms of probability distributions and in RNNs. They trained on approximately $2^1/_2$ hours of human and canine data. Although the study is focused on walking, rather than gestures, it is relevant nonetheless. The authors performed a perception-based evaluation, where

they found that the output of their model was rated highly 'natural', outperforming some other RNN or variational autoencoder based method, second only to ground truth motion. The distribution of produced poses, including mean and standard deviation, also matched the distribution of ground truth poses quite closely.

## 2.7 DeepDream and gradient-based explanation

An active area of machine learning research is explainability. Mechanisms of deep learning models are convoluted and models produce outputs without clues for their 'reasoning'. Gradient methods have long been used in explainability efforts [10, 99].

DeepDream, also called Inceptionism, is an tool for explaining CNNs, but became popular for generating eerie images [72, 73]. It is illustrated in figure 2.1. DeepDream is generally applied to image classification CNNs, initially to Inception [106].

DeepDream performs repeated gradient ascent. Unlike gradient descent happening during training in which model weights are adjusted, however, this optimisation is performed on the input image while model weights are held constant. The target of DeepDream is to increase the mean of the activations of certain layers within th CNN. It strives to make the CNN react more vigorously to the input image.

DeepDream is, in fact, part of a spectrum of methods that use the gradients of CNNs to generate images about their internal 'beliefs' of images, as presented in a well illustrated article [80]. These methods optimise the input image with the help of various forms of regularisation terms. They often start from random noise, and are able to produce images resembling desired classes (figure 2.2).

(a) Original photo

(b) Mean of internal activations of certain layers maximised

(c) Cross entropy loss of the output against the correct class minimised

Figure 2.1: DeepDream applied to an image of a dog. Gradient ascent is performed on the input image on the left to make it look even more like a dog. After a few iterations, this process results in the images on the right. The CNN believes even more strongly, that both (b) and (c) are Labradors, even more so than for (a). Images © Von.grzanka and Google, respectively. From `https://www.tensorflow.org/tutorials/generative/deepdream`.

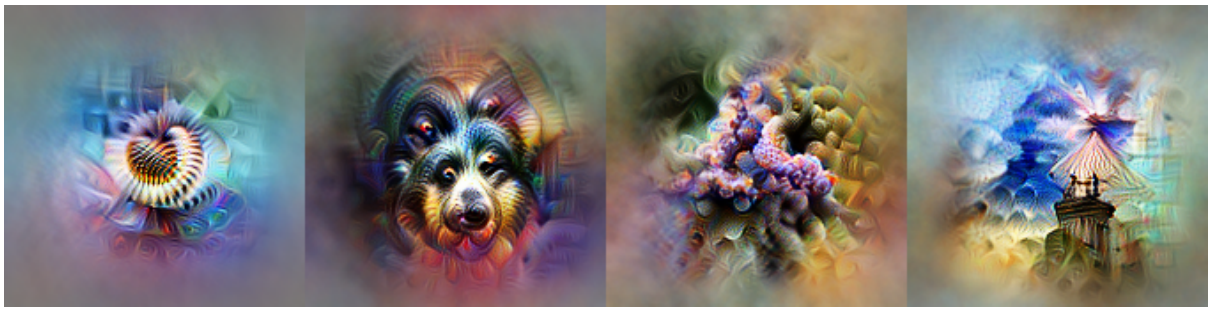


Figure 2.2: Input image optimised to increase the magnitude of activation of certain 'neurons' in a CNN. From [80] © Olah, Mordvintsev and Schubert, Google.

# 3 Design

My focus for the project changed significantly over the duration of the duration of the course. I describe the design for my initial project about body language in sections 3.1 and 3.2. Then, I detail my second project about 3D human pose estimation in section 3.3 on page 40.

## 3.1 Body language generation

I propose a method to generate body language using a Conditional GAN (CGAN) that takes text, voice, and personality into account. This GAN is trained on large amounts of video easily available of many individuals.

### 3.1.1 Model

The full block diagram of the body language GAN is shown on figure 3.1. The generator is an LSTM (figure 3.3) that outputs a 3D pose estimate at each timestep. It takes three inputs:

- **Audio features:** The audio of the speech provides the bulk of the timing and tonal information. It is passed through a fast-Fourier-transformer to obtain the log-power Mel-spectogram. This spectogram is processed using a set of 2D convolutions to compress it to into a single fixed-size audio feature vector per timestep. This is done similarly to [25]. A diagram of this CNN is shown in figure 3.2.
- **Word features:** To make use of the semantics information, the transcript of the speech is passed into GPT-2 [85], one of the state-of-the-art language modelling architectures. At each word, GPT-2 is allowed to see all of the words that come before (up to an arbitrary cutoff), but none that come after. Note, that GPT-2 itself is not trained or fine-tuned as part of this project, but is used as-is. The output of GPT-2 is used as latent features in a manner similar to how ResNet and VGG are used in the case of images (section 2.1.4 on page 7).

  Using Gentle [59], these words, and thus the word features, can be aligned to the

audio. The features from GPT-2 are repeated in every timestep for the duration of each word. A special token (empirically determined, such as $\mathbf{0}$ or $-\mathbf{1}$) is used for any periods of silence.

- **Latent input:** The latent input of the adjusts all the variation in body language that is feasible for the same text and voice (e.g. how wide, vigorous, or serious gestures are and what kind of gestures the speaker uses to emphasise their point). Therefore, this input is called **latent personality** for this project. It is a vector with its components randomly drawn from a Gaussian distribution. It is constant for each person and does not very with time.

Within the generator, the estimate from the previous timestep is also propagated as an input to the current timestep.

The discriminator is also an LSTM, shown in figure 3.4. It receives the audio and the word features like the generator, as these are the conditioning parameters of this CGAN. It also receives the pose estimates which it must evaluate. The source of these pose estimates is selected at random: they are either produced by the generator, or are estimated from the speech video and used as *pseudo* ground truth. The discriminator tries to predict at every timestep, whether it is seeing real or generated poses.

The Wasserstein loss setup is used for training [2], with an additional $L1$ term that gives helps the generator discover what gestures look like, while simultaneously ensuring that the generated gestures keep relatively high fidelity to the ground truth gestures. The models are trained using SGD, or a variant such as Adam [45].

### 3.1.2 Dataset

Deep learning and GANs in particular excel in regimes with massive datasets. Therefore, this body language generator is trained on as much video and as many individuals as possible.

TED is a popular series of short conference presentations of a diverse range of topics. It was selected as the initial source of the data, for several reasons:

- It is easily available through on internet [23].
- It is of high quality. Speakers are well prepared, speak clearly, and visibly gesticulate to enhance their point. Videos are curated, some are even manually transcribed.
- Large quantities are available. To date, approximately 30 days worth of TED talks have been filmed, presented by nearly 3200 unique individuals.

This dataset can be easily extended with the much larger but less curated TEDx [24], since it has a very similar format. TEDx has potentially 4000 years of video from more

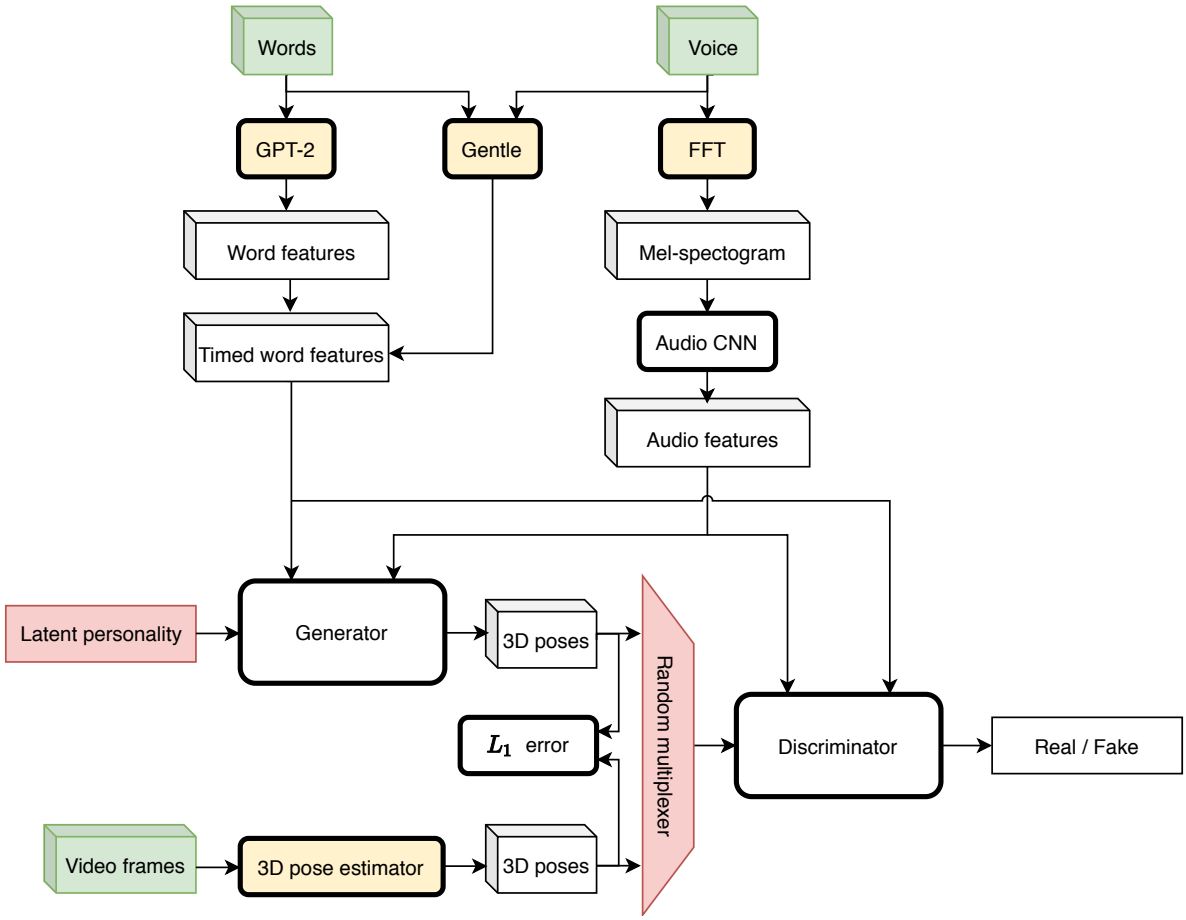



Figure 3.1: Architecture of the body language GAN. Boxes with depth represent time-series data. Green blocks are available from the ground truth dataset. Red blocks are randomly generated. Yellow rounded blocks are pre-trained tools that are used as-is, and are not themselves within the scope of this project.

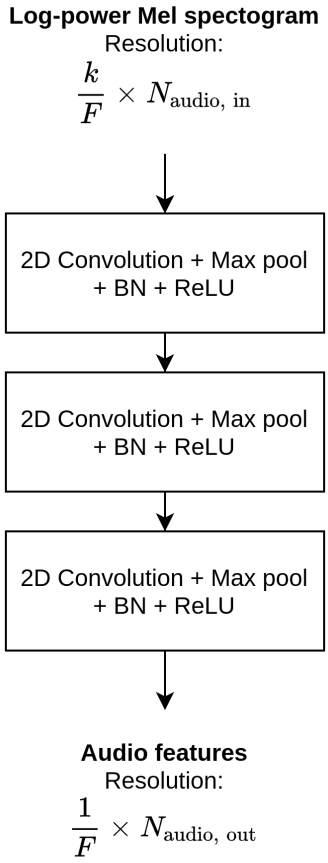



Figure 3.2: Audio CNN network design. $F$ is represents the length of a timestep, $k$ is a constant, depending on the sampling rate of the audio and the window size of the FFT. $N_{\text{audio, in}}$ is the number of Mel filterbanks produced by the FFT. $N_{\text{audio, out}}$ is the size of the audio features input into the GAN. The best output size, number of convolutional blocks, and number of channels within the convolutions can be found empirically. Note, that the convolutional kernels and the sizes of the pooling windows do not need to be square: rectangular kernels and windows may also be used in order to have different fields of view in the temporal and in the spectral dimensions and to arrived at the desired output size.

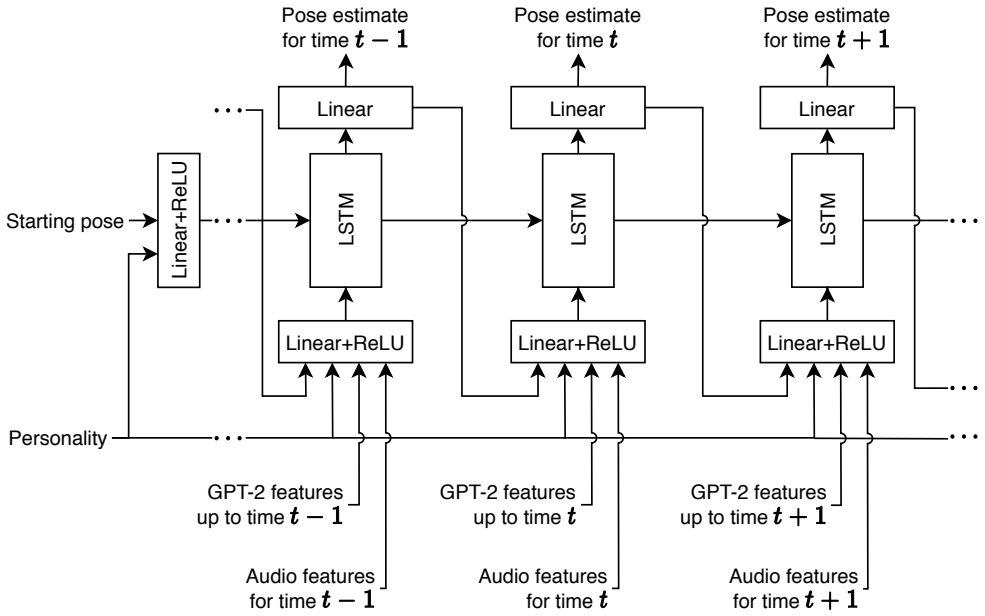



Figure 3.3: Architecture of the generator component of the body language GAN

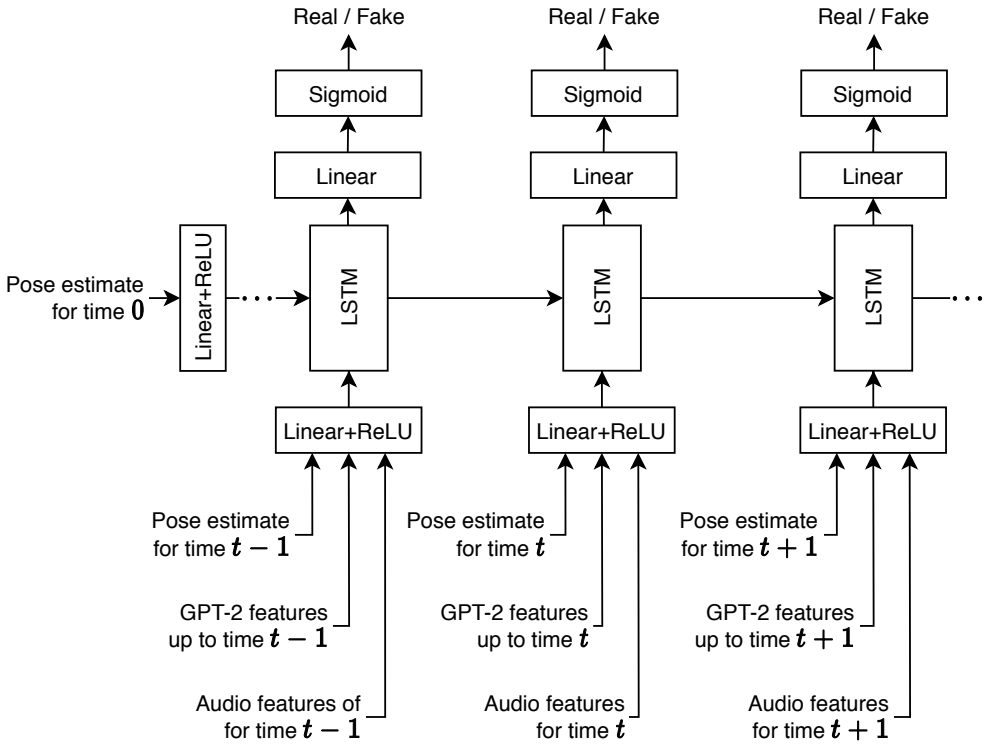



Figure 3.4: Architecture of the discriminator component of the body language GAN

than 150 000 speakers[1]. This is more than what could be processed with reasonable resources but the benefit gained by adding tens of thousands of further speakers likely also diminishes.

It is assumed that there is only one speaker in each video. Therefore, videos longer than 20 minutes are discarded.

Where transcripts are not available, automatically transcribed subtitles from YouTube or Google's Speech-to-Text transcription service [30] can be used. Using metadata from the video and naïve dictionary search, foreign-language presentations can be excluded.

The videos are cut into scenes using PySceneDetect [8], then passed into a state-of-the-art 3D pose estimator, that provides the pseudo-ground-truth pose of the speaker. Scenes that are below a threshold length, where a no pose is recognised, or where the pose is static (the slides are shown, rather than the speaker) are discarded.

The small portion of the remaining data is held out to form the validation set, while the rest can be used for training.

### 3.1.3 Potential variations

Several simple variations can be devised for this generator that can be implemented and tested using the same setup and may or may not alter the model's performance. The individual benefit of each of these modifications can be evaluated empirically using ablation studies.

**Context** is important for gestures. Some gestures include preparation [22], while others are influenced by words that follow. The base architecture presented in section 3.1.1 is causal, i.e. it only considers *past* context. Causality is advantageous in certain circumstances: in robotics or video game applications, the response may be improvised and the speech may only be synthesised seconds ahead of time or even less. Depending on the application, a compromise may be found by making the generator network look ahead by only a limited amount.

There are two straightforward ways to include future context in this architecture. First, the sizes of the convolution kernels in the Audio CNN can be tuned to increase or decrease the network's field of view in the temporal dimension. This method is useful to include *immediate* future context about the next few milliseconds. Second, the context can be given to the network explicitly. If $\Delta$ is the maximum look-ahead time permitted, input text up to $t + \Delta$ may be passed through a second GPT-2 block at time $t$. Features from

[1]There are over 150 000 videos on the TEDxTalks YouTube channel. The last $\sim 5000$ videos have an average duration of 13 minutes each.

this look-ahead GPT-2 can given to the generator as an additional, fifth input (not shown on figure 3.3).

**More latent input:** Constraining the latent input to the generator to be constant with time could be too restrictive. An addition latent term could be added as input to the generator, that changes on each timestep. This could better account for the full extent of the gesture distribution even better. However, it would probably also somewhat undermine the body language understanding project (section 3.2 on the following page).

**Dropout** [101] (also used by Yoon et al. and Ferstl et al.) and input dropout in particular is a technique that can improve the robustness of networks to limited amounts of erroneous data. In GANs, it can also form a supplementary latent input [117]. It can be expected to have a beneficial effect to the performance, though the noise in the generator could possibly also degrade naturalness.

**Attention** [5] (also used by Yoon et al.) is a simple addition to the network. It can emphasise inputs from timesteps it learned as important and diminish the significance of timesteps that are less useful, making a more informed estimate. It would likely improve the performance of the generator and the discriminator with little additional computation.

**More adversaries:** Ferstl et al. presented a GAN that used several adversaries to train its generator. The 'motion realism discriminator' is already part of the current model. Other adversaries, such as the 'minibatch discriminator' and the 'displacement discriminator' are easy and useful additions. Furthermore, it has been shown in [71], that using an ensemble discriminators with the same architecture can further improve performance.

### 3.1.4 Discussion

Yoon et al. [124] also used TED talks as their training dataset. However, they discarded the vast majority of their data and could only use some ideal scenes. This was caused, by the lack of a reliable pose estimation tool. As it turns out, I was not able to circumvent this issue either. However, using only 13 % of the data would have a strong adverse effect on the performance my project: all available data must be utilised efficiently in order to train the latent personality accurately and make the second part of the project (section 3.2) feasible.

Multiple other authors attempted using GANs for generating human pose sequences [22, 25]. However, they only used textual or audio input to the generator, not both.

Furthermore, they omitted the latent input, which helps GANs generate non-deterministic outputs and learn output *distributions*.

Some studies ([124] and [22]) include future context using a *bidirectional* LSTM [31]: an architecture that has a second LSTM sequence, starting at the end of the sentence and working backwards. This approach sacrifices the causality of the network and makes it more complicated to generate poses to improvised inputs.

Some works use an $L_2$ loss as a guide for the pose generator [22, 51, 124], while others use $L_1$ [25]. Both have been shown to be effective for the regression of human poses. However, $L_2$ penalises large deviations from the target much more heavily than $L_1$, which penalises large and small deviations equally. This is undesirable in a setting that tries to provide the generator some flexibility in the exact pose. $L_1$ is also shown to work better in pose estimation [104].

The dataset used can also be extended to different data sources, such as lectures, news, or talk shows. However, it is possible that that would be detrimental to the performance of the model, as these videos have a different presentation format and the speakers may gesticulate somewhat differently.

An improved policy to include videos that show a series of speakers sequentially would be to use a state-of-the-art face matching CNN to identify and potentially re-identify speakers. Face matching would also be useful if videos with more than one individual were used, studying gestures of conversation. However both of these improvements are outside the scope of this project.

Unlike BERT [16] used in the contemporary Gesticulator [51], GPT-2 is *autoregressive*, which means that it can be easily applied word by word, without looking ahead beyond the current word. This is important, for robotics applications, as the text may not be known very much ahead of time. GPT-2 is also trained on a much larger text corpus.

## 3.2 Body language understanding

By training the GAN on a large amount of data to synthesise body language, the network is also trained to have a good understanding of gesture space and the correlation of personality and body language. The generator part may be used for a secondary purpose: to understand body language present in videos. The technique for this highly resembles the gradient methods discussed in section 2.7 on page 28 and it is illustrated in figure 3.5.

The generator is repeatedly evaluated on *small segments* of a speech. These segments may be part of the training set. From these, the generator produces 3D pose sequences.

The $L_1$ error compared to the pseudo ground truth pose sequence is computed. Using gradient descent, the latent personality is adjusted to diminish this $L_1$ error, while keeping all model parameters constant (frozen).

Some additional terms may be beneficial. Maximise intermediate activations of the network (like DeepDream) may produce more *typical*, emphasised results. It may also necessary to add a regulariser term that keeps the latent personality within a likely region of the latent personality distribution.

The latent personality is initialised for gradient descent with the latent personality used for training the generator if the video clip is part of the training set, or with an arbitrary random vector otherwise.

The above procedure produces latent personality values that are most likely to gesticulate with the exact gestures seen in the video *segment*, according to the perception of the network. By analysing changes over time of this inferred personality for the same individual, their behaviour can be studied. Are they nervous or excited at the beginning? Do they become more jovial at some part of the speech? How do they react to the clap of the audience? Similar questions can be answered by correlating the inferred latent personalities with latent personalities of real individuals.

The inferred latent personality can be analysed by calculating its distance to latent personalities of a few, manually selected individuals. However, these vectors are expected to carry a substantial amount of noise, making analysis difficult. A more advanced approach to this comparison is to train a small DNN classifier on latent personalities of *few* manually selected real individuals. For example, by selecting a few dozen generally jovial speakers and a few dozen serious speakers, a classifier can place an inferred personality on this spectrum.

The generator is trained in section 3.1 with a constant latent personality over time for each individual. The speaker has slight variations and adjustments in body language, but does not change character completely. The latent personality used for training is the *mean* character of the individual over the few minutes of the video.

Gradient optimisation is a slow and computationally expensive method. The target of this technique is not real-time use. Instead, studying reactions and adjustments in character of real individuals can pave the way to create improvising robotic systems that adjust some parameters of their personality on-the-fly to appear more natural.

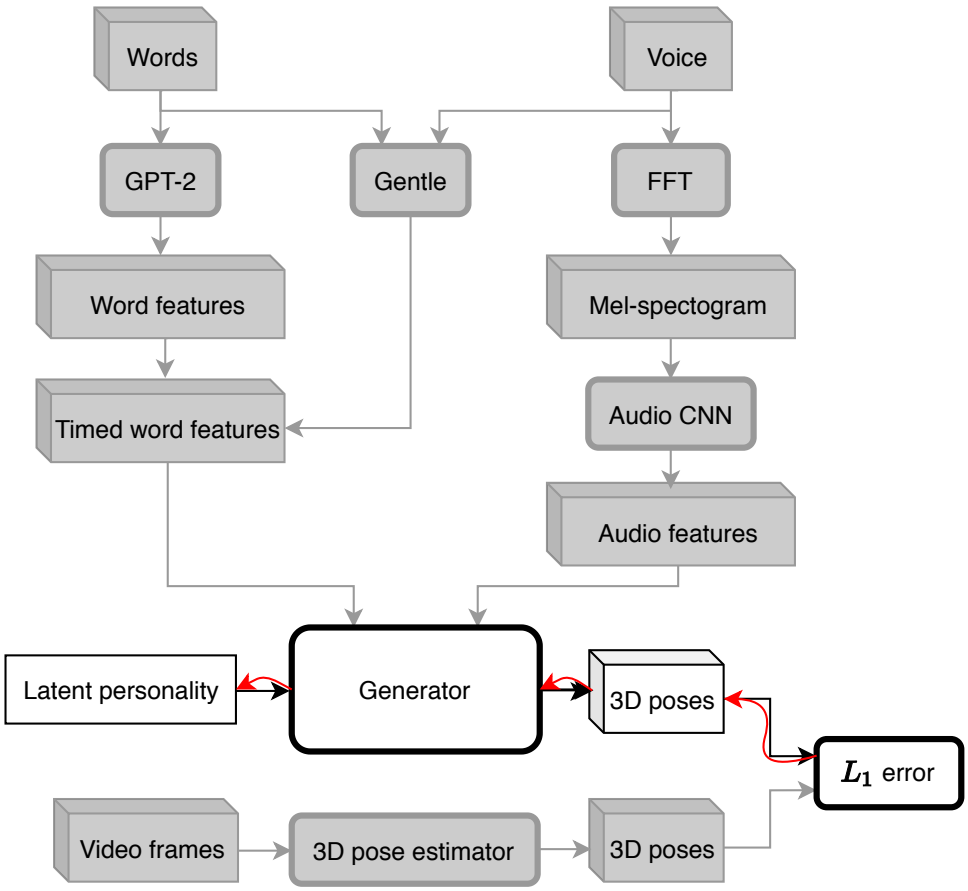



Figure 3.5: Block diagram of the body language understanding model. It is simply an alteration the body language generation architecture from figure 3.1. Boxes with depth represent time series data. Grey blocks are only evaluated forwards, gradients are not propagated back through them. Red arrows show the propagation of the gradient.

## 3.3 3D pose estimation

After I recognised that the project in sections 3.1 and 3.2 is infeasible due to the lack of a reliable 3D pose estimator, I shifted my attention to this issue.

### 3.3.1 Dataset

The pose estimator is trained on as much data as available to improve its generalisation to unseen scenarios. Primarily, it is trained on Human3.6M [40], Total Capture [110], Total Motion dataset [120], and MPII-3DHP [67]. These are all mo-cap based indoors datasets with 7.7 million frames of 56 actors and actresses from 4 different rooms and lighting conditions.

The 2D COCO pose dataset is also used from semi-supervised training to improve in-the-wild accuracy.

#### Data augmentation

Data augmentation is a crucial part of this design aiming to be robust to a wide range of scenarios and imperfections.

**Texture:** MPII-3DHP is filmed in a way permitting the augmentation of the background and the actors foreign textures. A random variation of textures are applied in this design, as recommended by [67].

**Random crop, rotate & flip** is a standard data augmentation technique used by many CNNs. Input images are rotated with a small but random angle (e.g. $-45°$–$45°$) and then cropped to a random size such that a at least a few joints of the person is still visible. Unlike some other works, this setup allows parts of the person to be cut off. Specifically, the dataset is augmented with images, where the legs and/or one arm have been cropped and only the bust is visible.

The pose labels are transformed accordingly. Any joint that ends up outside the image after the transformations is marked as such, so that its position error can be discarded. Furthermore, the image and the pose are flipped around the vertical axis with a 50 % probability.

**Occlusion and self-occlusion:** Small parts of the image are hidden by rectangles of random colours to make the model robust against occlusion by objects. Any joint that falls under a rectangle is marked as occluded.

Joints in Total Motion are marked for self-occlusion. For the other three 3D datasets, the ground truth 3D poses and the 'cylinder man' model from [12] are used to mark self-occluded joints.

### 3.3.2 Model architecture

The model architecture is made up of two main phases: the backbone and the heads.

EfficientNet [107] is used as a backbone. The architecture taps into the activations at the output of stages 5 and 7 of EfficientNet. Resolution of the latter stage is increased using a transposed convolution to equal the resolution of the former stage. The two are concatenated in the channel dimension, and resolution is increased once again with another transposed convolution. The result forms the input to the heads.

There are three heads used in collaboration in this design. All of them use the stages laid out by OpenPose [6]. This provides sufficient complexity and opportunity for intermediate supervision.

The joint head consists of two OpenPose stages. It outputs a 2D heatmap and a 1D depth map for each joint. Together, this is $(2 + 1) \times |\text{joints}|$ output channels. The heatmap estimates the likelihood that the respective joint is at under a pixel, while the depth map estimates the dept of that joint, relative to the skeleton root.

The spatial relation head consists of four OpenPose stages. It outputs a 3D vector field for each pre-defined joint pair, thus requiring $3 \times |\text{joint pairs}|$ output channels. It predicts the vector to pointing to the location of the second joint in the pair given that the first joint is located at the current pixel. A the most straightforward set of pairs is parent $\rightarrow$ child.

The temporal relation head is a little more involved. It considers a set of frame pairs. For illustration, a suitable set of pairs is $\{(-8, 0),\ (-4, 0),\ (-2, 0),\ (-1, 0),\ (1, 0),\ (2, 0),\ (4, 0),\ (8, 0)\}$, where the numbers represent the distance in frames from the 'currently processed' frame. This allows several relevant scales of motion to contribute to the estimate. However, this can be adjusted based on the maximum permissible temporal field of view. Pairs with positive indices can be dropped if a causal system is required. First, the backbone outputs from the last transpose convolutions of the two frames in a each pair are concatenated in the channel direction. Then, four OpenPose stages are computed on the combined features, which output a 3D vector field for each joint and temporal pair (i.e. $3 \times |\text{joints}| \times |\text{temporal pairs}|$ channels). They are trained to predict the displacement of the *same* joint between the two time steps of the pair.

The next step is to combine the information inferred in these heads into a pose estimate. This is done by combining the separate estimates of each head.

The spatial estimate of the joint head is the spatial expectation of the heatmap. The depth estimate is the expectation of the depth map, weighed by the heatmap.

For the estimates of the relation heads, the vector fields in each relation are weighted by the heatmaps of the parent joint in the relation. The expectation is calculated, resulting in an estimated vector displacement for the relation. To form a location estimate for the head, this displacement is added to the estimated location of the parent joint in the relation. The root joint is defined to have a depth of 0.

The final estimate is a linear combination of the estimate the joint, the spatial, and each each temporal head.

#### Training losses

The joint depth, spatial, and temporal relation heads are trained explicitly using the ground truth vectors of their respective relations. Heatmap heads are trained implicitly, through the regression loss [104].

Heatmap pixels near a joint have a stronger clue about the location of the joint than pixels further away. In order to train the networks to produce localised heatmaps, the Jensen-Shannon divergence of the heatmaps from a Gaussian distribution heatmap, centred at the ground truth location and a variance of 1 pixel is added to the loss [77].

Weak supervision is applied with a small weighting, penalising the changes in ratios of bones.

Intermediate supervision has been shown to have a strong beneficial effect on accuracy. Accordingly, multiple identical OpenPose modules are given the same target and are included in the loss. They are expected to provide more and more refined estimates.

Terms due to joints that are not visible on the image (occluded or self-occluded) are not included in the final loss, as it was shown in [12] that including them has a significant detrimental effect to accuracy.

### 3.3.3 Discussion

**Data augmentation** beyond basic random rotation, cropping, and flipping is not performed in most works, because they are expected to perform well only on the validation set of full-body datasets. It is, however, crucial in a tool that is expected to work in a variety of scenarios.

**EfficientNet:** ResNet [33] and VGG [100], used in the majority of transfer learning based CNNs, were proposed in 2016 and 2014, respectively. Significant progress has been made since. EfficientNet can produce better features for an order of magnitude

less computation. While it has some academic value to compare works using the same backbone, it is desirable to use the state of the art when building a tool for accuracy.

**Heads:** Several near-state-of-the-art pose estimator works (e.g. [104] and [103]) include exciting innovations, but are likely limited by the head they use: simply two transpose convolutions following the backbone, like Simple Baselines [121]. This is a decidedly simple architecture for 2D pose and may not be able to encode a sufficient complexity for accurate 3D pose estimation. Accordingly, the presented design uses more convolutions in the form of OpenPose blocks following the transpose convolutions, allowing it to arrive at a more precise estimate.

[104] has shown, that using the expectation of a likelihood map instead of the mode enables it to consider more of the information encoded in the map and return a better estimate. Therefore, expectation is used extensively in this implementation.

The overall output structure was most influenced by ExplicitPose [103]. The use of OpenPose blocks is similar to Monocular Total Capture [120], which uses the older Convolutional Pose Machine block [116].

# 4 Evaluation

## 4.1 Body Language

### 4.1.1 Progress to date

#### Dataset

I began the project by assembling the dataset. Using a simple script and the youtube-dl project [125], I downloaded over 100 days worth of TED and TEDx talks.

#### Pseudo ground truth poses

I then evaluated several 3D pose estimation tools. Due to the unmaintained nature of these projects, it took days for each to fix all version issues and bugs and figure out, how to run them. Here I discuss my experience the most relevant projects I attempted.

There are two main issues generally encountered: inconsistent poses and jitter. Inconsistent poses happen when the estimator confuses parts of two individuals or of the same individual. This is less common. Jitter is a more common issue, present in nearly all estimates, particularly when the scene is not ideal. In this error mode, the estimator estimates the rough area of joint correctly, but misses the exact joint location slightly. At the scale of the skeleton, this is an error on the order of a a few centimetres. However, the direction of the error varies from frame to frame, based on small random details and noise in the input frame. When the estimators are applied to videos, these inconsistent errors create a sense of shaking and make it more difficult to discern gestures purely from the resulting skeleton.

Though an interesting idea, Margipose [77] was not able to give stable estimates. It moved a lot between frames and struggled significantly both with cropped bodies and people in the background. Significant jitter is also present.

Hossain et al. reported good results in their paper [37]. However, pose estimates appear unreliable and unstable in their demo video.

OpenPose [6] and AlphaPose [20] produced good estimates of the 2D poses. Jitter was still present, but not as much as in 3D pose estimators. However, these tools do not output depth information, rendering the body language GAN unable to learn the full 3D skeleton, making generated poses more difficult to remap to a robot or a 3D model, resulting in a less useful project.

VideoPose3D [82] has looks very smooth in demos. However, I soon discovered that it only works on full body images. This is likely because it does not take the image, occlusion, or joint confidences into account in its convolutions. When part of the body is occluded, the rest of the joint estimates become unreliable too.

Monocular Total Capture [120] was the most promising tool due to its amazingly stable demo video. Unfortunately, it rendered all-black frames once I managed to get it running, which I was not able to fix. However, it became apparent that it is a very slow method, taking 3 hours for 20 second video segments. This is too slow for a large dataset, making the method infeasible.

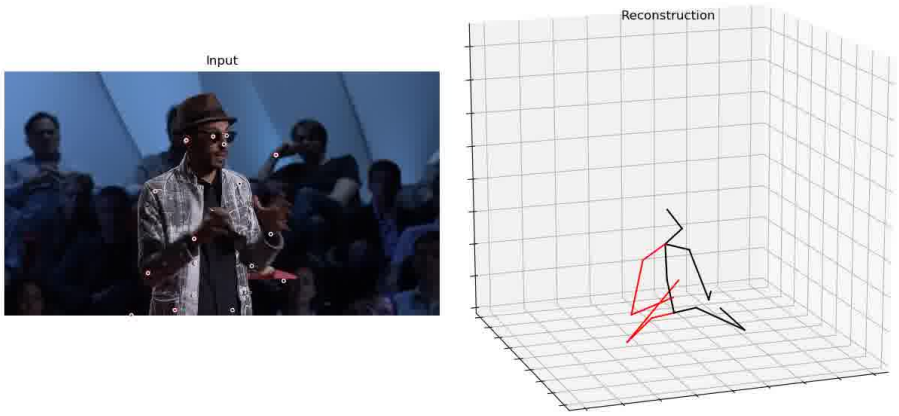
Unfortunately, there was no code published for ExplicitPose [103] and Occlusion-Aware Networks [12], therefore I could not evaluate them.

### 4.1.2 Metrics

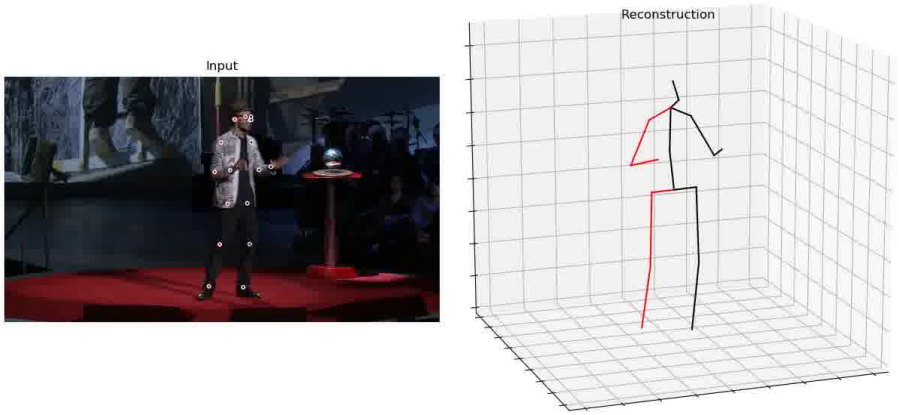
The discriminator of a GAN can be evaluated by computing, what percentage of its input it is able to correctly classify as real or fake. The generator can be evaluated by computing what percentage of its outputs manages to fool the discriminator. By withholding a small portion of the dataset from training, the generalising ability of the discriminator can also be validated.

A simple evaluation strategy to measure the similarity between th ground truth and the generated gestures is to cut up the gesture into smaller, sub-second chunks. These chunks can be considered rigid-bodies and they can be slid/stretched in time *slightly* to create a better alignment without reordering (for example through Procrustes analysis). Similarity between pose sequences can be measured after the realignment. This gives a more robust estimate than a simple $L_1$ difference. In a way this metric is similar to PA-MPJPE of 3D pose estimation.

To further evaluate the generator, a user study must be conducted, similarly to [25] or [124]. A good venue for this is Amazon Mechanical Turk. Subjects watch a stick figure perform the gestures generated by the model, along with the audio. They are then asked to grade the performance on several axes, loosely following [124], such as: Natural–Fake, Artificial–Lifelike, Rigid–Elegant, Like–Dislike, Jovial–Serious, Extroverted–Introverted or gesture-speech correlation. They model can be compared against random and nearest-




(a) Estimate from VideoPose3D in a cropped setting. While the incorrect legs could be removed through a combination with 2D methods, this setting also impacts the accuracy and smoothness of other joints.

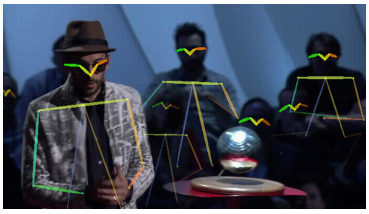



(b) A near perfect estimate from VideoPose3D in a rare, ideal scene

(c) Margipose is not robust to images that do not include the entire body. It also has a limited field of view.



(d) SPIN [47] (not listed on the previous page) is also not robust to cropped images.



(e) A good 2D only estimate from AlphaPose. Some jitter is still visible in a video.

Figure 4.1: Example frames from the outputs of some of pose estimators available online

neighbour baselines. These baselines can also be used to filter out uncommitted subjects. By varying the latent personality, its effect can be quantified

If Stevie [60] is available, particularly if he gains lower arms, joints can be remapped and he can also be used to augment the evaluation process. Subjects watch a speech performed by Stevie, that is spoken by a famous person, with that person's face on Stevie's LCD. The gestures performed are either the real gestures used, another person's gestures, or generated gestures. Subjects are asked to rate, whether they believe it was the real speaker's body language.

The inferred latent personality is likely to be very noisy. Therefore a frequency space analysis, looking for low frequency variations is necessary. Furthermore, the data characterising of latent personalities in the Amazon Mechanical Turk study can be used to train a small DNN that classifies latent personalities along some of the above spectra. Running this DNN on the inferred personalities and observing changes in character along time is the main form of evaluation and the target of the body language understanding project.

## 4.2 3D pose estimation

### 4.2.1 Progress to date

Throughout the project, I used Python and the Pytorch deep learning framework [81]. I built an easily configurable framework that allows for different backbones and heads. I built a dataset reader for the Total Motion and the Human3.6M datasets. I implemented all necessary components of the network and the main loss functions. Weak supervision and intermediate supervision are not yet implemented. I created some tools to visualise heatmaps, vector fields and pose estimates generated by the network. However, not all bugs have been fixed yet, resulting in inaccurate outputs.

### 4.2.2 Metrics

The model is evaluated on 3D datasets using the standard MPJPE metric. PA-MPJPE is also evaluated, but that is not the main focus of the project. In Procrustes analysis, the skeleton is realigned on every frame, presenting for an opportunity to absorb some of remaining the jitter.

Calculating MPJPE on in-the-wild images and videos is not possible, as 3D ground truth is not available. The 2D pose can be evaluated on 2D datasets using the OKS and the AP metrics.

The weak supervision signal can also be used, computing the variance of the ratio between

lengths of various pairs of joints of all people in the dataset, and also of a set of distinct individuals. The TED dataset from the gesture generation project may be practical for this purpose.

One of the goals of this project is to create a tool that is robust to scenarios like cropped images or occluded joints. Accordingly, the tool is also evaluated, when fed the same random augmented data as used for training in section 3.3.1.

The temporal stability is also evaluated on videos from the 3D datasets. The velocity of a joint can be calculated by taking the difference between frames. Its acceleration can be calculated using second order differences, i.e. using the difference between adjacent velocities. To evaluate the temporal stability, the mean and variance of the *difference* between the estimated and the ground truth velocities and accelerations. These measures characterise the *erroneous* velocity and acceleration.

Another way to evaluate jitter, which can be done without ground truth poses, is to modify the input frames. These modifications can be diverse, such as: an imperceptible amount of additive Gaussian white noise added to the image, shits of only a few pixels, or rotations of only a few degrees. The network will react to these conditions slightly differently each time, showing any frailties. The jitter can be quantified by the variance of these estimates.

# 5 Conclusion

The contributions of this work are threefold:

1. It proposes an architecture to learn character-dependent body language, that is in line with progress from previous research. It also provides a brief survey of this research and some novel methods for evaluating the architecture's performance. The task requires enormous amounts of data, particularly to make sub-project 2 feasible, but an easily available source of this data is also pointed out. This study shows the opportunity present in the quantities of data available today. It also recognises and details the limitations of currently available tools that prevent the learning from being carried out to its full intended extent.

2. It presents a way to solve the inverse problem of learning character-dependent body language: inferring character based on body language. This is done without any explicit supervision data being available for it. Rather, it takes advantage of a well trained model from the first sub-project. To the best of my knowledge, such a project has not been attempted before. Getting this to work well can pave the way to creating improvising and character-adjusting agents in video games and robots. For this it requires high quality tools to extract robust data from videos and a good model for sub-project 1.

3. After recognising a gap between the required and available 3D pose estimation tools, a method is presented for training a more robust tool by incorporating good practices from state-of-the-art studies. These practices and studies are summarised in a short survey. This work also proposes a set of evaluation measures that can not only compare it to other works, but measure, weather the targeted increase in robustness and joint stability has been achieved.

The outset of this work targeted teaching Stevie social skill, focusing on body language. While the first and and particularly the second sub-project initially seemed to be on the edge of what is feasible, it turned out to be extremely ambitious, perhaps too much so for the constraints of this course. This is in part, due to the overestimation of the performance state of the art of 3D human pose estimators. 3D pose estimation was taken

as a nearly solved problem, but it was shown that more work needs to be done. The project was re-targeted to address this issue and significant progress has been made in the very limited time frame that was left as part of this course. More work remains to be done before the issue can be considered solved.

## 5.1 Future work

### 5.1.1 Body language

The project was hindered by the lack of a robust and stable 3D pose estimator. Once such a tool is ready, the GAN can be trained and evaluated. Evaluating, how capable the architecture is of learning idiosyncrasies and using it in further projects that enable improvisation and adjustment of character would be an exciting outcome.

While RNNs, including LSTMs, have shown excellent performance on sequential data, they have more recently been outperformed by Transformer-based non-recurrent models in language modelling [16, 85, 113]. Dilated temporal convolutional networks, demonstrated in VideoPose3D [82], for example, have also shown surprisingly good performance. It would be an interesting direction for future work to try out these architectures for human pose generation.

Normalising flows, used successfully in [35], is a recent innovation that excels at learning probability distributions in a self-supervised manner. It is a machine learning framework, that is computed somewhat differently, compared to deep learning and GANs. Since this project tries to learn a probability distribution of gestures and personalities, porting it to normalising flow would be an exciting future path.

### 5.1.2 Pose estimation

Most important for the pose estimator is to bring it in line with the design laid out in section 3.3. It must be quantitatively evaluated, then published so that other researchers and engineers can benefit from it. Extending it to also include hand coordinates would be relatively simple, but highly beneficial. Extending the estimator for multi-person estimation and pose-root estimation are non-trivial, but existing work details, how it can be done.

It would be an interesting way of evaluation to train a small CNN on the outputs of specific joints. This CNN is optimised to predict the error the estimator will make in its estimate, based on a small cropping around the joint. This could demonstrate, how much of the jitter is purely random, and how much is a deficiency in the model.

Inference time optimisation, though results in stable images, is a very expensive

operation and must be avoided for a near-real-time application. However, studies that used it so far have relied on inefficient, single-core CPU implementations. It would be interesting to try an efficient GPU implementation, and compare the accuracy vs. compute time trade-off.

# Bibliography


[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler and Bernt Schiele. '2D Human Pose Estimation: New Benchmark and State of the Art Analysis'. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2014.

[2] Martin Arjovsky, Soumith Chintala and Léon Bottou. 'Wasserstein GAN'. In: (2017). arXiv: 1701.07875 [stat.ML].

[3] Anurag Arnab, Carl Doersch and Andrew Zisserman. 'Exploiting Temporal Context for 3D Human Pose Estimation in the Wild'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019). DOI: 10.1109/cvpr.2019.00351. URL: http://dx.doi.org/10.1109/CVPR.2019.00351.

[4] B. Babagholami-Mohamadabadi, A. Jourabloo, A. Zarghami and S. Kasaei. 'A Bayesian Framework for Sparse Representation-Based 3-D Human Pose Estimation'. In: *IEEE Signal Processing Letters* 21.3 (Mar. 2014), pp. 297–300. ISSN: 1558-2361. DOI: 10.1109/LSP.2014.2301726.

[5] Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. 'Neural Machine Translation by Jointly Learning to Align and Translate'. In: (2014). arXiv: 1409.0473 [cs.CL].

[6] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei and Yaser Sheikh. 'OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields'. In: *arXiv preprint arXiv:1812.08008*. 2018.

[7] Zhe Cao, Tomas Simon, Shih-En Wei and Yaser Sheikh. 'Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields'. In: *CVPR*. 2017.

[8] Brandon Castellano. *PySceneDetect. Python and OpenCV-based scene cut/transition detection program & library*. 2014. URL: https://github.com/Breakthrough/PySceneDetect (visited on 03/05/2020).

[9] Cristian Sminchisescu Catalin Ionescu Fuxin Li. 'Latent Structured Models for Human Pose Estimation'. In: *International Conference on Computer Vision*. 2011.

[10] Aditya Chattopadhay, Anirban Sarkar, Prantik Howlader and Vineeth N Balasubramanian. 'Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks'. In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2018, pp. 839–847.

[11] Ching-Hang Chen and Deva Ramanan. '3D Human Pose Estimation = 2D Pose Estimation + Matching'. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[12] Yu Cheng, Bo Yang, Bo Wang, Wending Yan and Robby T. Tan. 'Occlusion-Aware Networks for 3D Human Pose Estimation in Video'. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2019.

[13] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau and Yoshua Bengio. 'On the Properties of Neural Machine Translation: Encoder–Decoder Approaches'. In: *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (2014). DOI: 10.3115/v1/w14-4012. URL: http://dx.doi.org/10.3115/v1/W14-4012.

[14] Matthias Dantone, Juergen Gall, Christian Leistner and Luc Van Gool. 'Body parts dependent joint regressors for human pose estimation in still images'. In: *IEEE transactions on pattern analysis and machine intelligence* 36.11 (2014), pp. 2131–2143.

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei. 'Imagenet: A large-scale hierarchical image database'. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. 'BERT: Pre-training of deep bidirectional transformers for language understanding'. In: *arXiv preprint arXiv:1810.04805* (2018).

[17] Jeff Donahue, Philipp Krähenbühl and Trevor Darrell. 'Adversarial Feature Learning'. In: (2016). arXiv: 1605.09782 [cs.LG].

[18] Vincent Dumoulin and Francesco Visin. 'A guide to convolution arithmetic for deep learning'. In: *ArXiv e-prints* (Mar. 2016). eprint: 1603.07285.

[19] Marcin Eichner, Manuel Marin-Jimenez, Andrew Zisserman and Vittorio Ferrari. '2d articulated human pose estimation and retrieval in (almost) unconstrained still images'. In: *International journal of computer vision* 99.2 (2012), pp. 190–214.

[20] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai and Cewu Lu. 'RMPE: Regional Multi-person Pose Estimation'. In: *ICCV*. 2017.

[21] V. Ferrari, M. Marin-Jimenez and A. Zisserman. 'Progressive search space reduction for human pose estimation'. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. June 2008, pp. 1–8. DOI: 10.1109/CVPR.2008.4587468.

[22] Ylva Ferstl, Michael Neff and Rachel McDonnell. 'Multi-objective adversarial gesture generation'. In: *Motion, Interaction and Games*. 2019, pp. 1–10.

[23] TED Foundation. *TED – YouTube*. 2020. URL: https://www.youtube.com/user/TEDtalksDirector (visited on 05/05/2020).

[24] TED Foundation. *TEDx Talks – YouTube*. 2020. URL: https://www.youtube.com/user/TEDxTalks (visited on 05/05/2020).

[25] Shiry Ginosar, Amir Bar, Gefen Kohavi, Caroline Chan, Andrew Owens and Jitendra Malik. 'Learning individual styles of conversational gesture'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 3497–3506.

[26] Rohit Girdhar, Georgia Gkioxari, Lorenzo Torresani, Manohar Paluri and Du Tran. 'Detect-and-Track: Efficient Pose Estimation in Videos'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (June 2018). DOI: 10.1109/cvpr.2018.00044. URL: http://dx.doi.org/10.1109/CVPR.2018.00044.

[27] Ross Girshick, Ilija Radosavovic, Georgia Gkioxari, Piotr Dollár and Kaiming He. *Detectron. FAIR's research platform for object detection research, implementing popular algorithms like Mask R-CNN and RetinaNet*. 2018. URL: https://github.com/facebookresearch/detectron.

[28] Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep learning*. MIT press, 2016.

[29] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville and Yoshua Bengio. 'Generative adversarial nets'. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.

[30] Google. *Cloud Speech-to-Text – Speech Recognition | Google Cloud*. 2020. URL: https://cloud.google.com/speech-to-text (visited on 05/05/2020).

[31] Alex Graves and Jürgen Schmidhuber. 'Framewise phoneme classification with bidirectional LSTM networks'. In: *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.* Vol. 4. IEEE. 2005, pp. 2047–2052.

[32] Kaiming He, Georgia Gkioxari, Piotr Dollar and Ross Girshick. 'Mask R-CNN'. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct. 2017). DOI: 10.1109/iccv.2017.322. URL: http://dx.doi.org/10.1109/ICCV.2017.322.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. 'Deep residual learning for image recognition'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. 'Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification'. In: *2015 IEEE International Conference on Computer Vision (ICCV)* (Dec. 2015). DOI: 10.1109/iccv.2015.123. URL: http://dx.doi.org/10.1109/ICCV.2015.123.

[35] Gustav Eje Henter, Simon Alexanderson and Jonas Beskow. 'MoGlow: Probabilistic and controllable motion synthesis using normalising flows'. In: (2019). arXiv: 1905.06598 [cs.LG].

[36] Sepp Hochreiter and Jürgen Schmidhuber. 'Long Short-Term Memory'. In: *Neural Comput.* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[37] Mir Rayat Imtiaz Hossain and James J. Little. 'Exploiting Temporal Information for 3D Human Pose Estimation'. In: *Lecture Notes in Computer Science* (2018), pp. 69–86. ISSN: 1611-3349. DOI: 10.1007/978-3-030-01249-6_5. URL: http://dx.doi.org/10.1007/978-3-030-01249-6_5.

[38] Peiyun Hu and Deva Ramanan. 'Bottom-Up and Top-Down Reasoning with Hierarchical Rectified Gaussians'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016). DOI: 10.1109/cvpr.2016.604. URL: http://dx.doi.org/10.1109/CVPR.2016.604.

[39] Sergey Ioffe and Christian Szegedy. 'Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift'. In: (2015). arXiv: 1502.03167 [cs.LG].

[40] Catalin Ionescu, Dragos Papava, Vlad Olaru and Cristian Sminchisescu. 'Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (July 2014), pp. 1325–1339.

[41] Sam Johnson and Mark Everingham. 'Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation'. In: *Proceedings of the British Machine Vision Conference*. doi:10.5244/C.24.12. 2010.

[42] Hanbyul Joo, Tomas Simon, Xulong Li, Hao Liu, Lei Tan, Lin Gui, Sean Banerjee, Timothy Scott Godisart, Bart Nabbe, Iain Matthews, Takeo Kanade, Shohei Nobuhara and Yaser Sheikh. 'Panoptic Studio: A Massively Multiview System for Social Interaction Capture'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2017).

[43] Hanbyul Joo, Tomas Simon and Yaser Sheikh. 'Total capture: A 3d deformation model for tracking faces, hands, and bodies'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8320–8329.

[44] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen and Timo Aila. 'Analyzing and Improving the Image Quality of StyleGAN'. In: (2019). arXiv: 1912.04958 [cs.CV].

[45] Diederik P. Kingma and Jimmy Ba. 'Adam: A Method for Stochastic Optimization'. In: (2014). arXiv: 1412.6980 [cs.LG].

[46] Ivan Kobyzev, Simon J. D. Prince and Marcus A. Brubaker. 'Normalizing Flows: An Introduction and Review of Current Methods'. In: (2019). arXiv: 1908.09257 [stat.ML].

[47] Nikos Kolotouros, Georgios Pavlakos, Michael Black and Kostas Daniilidis. 'Learning to Reconstruct 3D Human Pose and Shape via Model-Fitting in the Loop'. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)* (Oct. 2019). DOI: 10.1109/iccv.2019.00234. URL: http://dx.doi.org/10.1109/ICCV.2019.00234.

[48] Sven Kreiss, Lorenzo Bertoni and Alexandre Alahi. 'PifPaf: Composite Fields for Human Pose Estimation'. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019). DOI: 10.1109/cvpr.2019.01225. URL: http://dx.doi.org/10.1109/CVPR.2019.01225.

[49] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. 'Imagenet classification with deep convolutional neural networks'. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[50] Taras Kucherenko, Dai Hasegawa, Gustav Eje Henter, Naoshi Kaneko and Hedvig Kjellström. 'Analyzing Input and Output Representations for Speech-Driven Gesture Generation'. In: *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents* (July 2019). DOI: 10.1145/3308532.3329472. URL: http://dx.doi.org/10.1145/3308532.3329472.

[51] Taras Kucherenko, Patrik Jonell, Sanne van Waveren, Gustav Eje Henter, Simon Alexanderson, Iolanda Leite and Hedvig Kjellström. 'Gesticulator: A framework for semantically-aware speech-driven gesture generation'. In: *arXiv preprint arXiv:2001.09326* (2020).

[52] Harold W Kuhn. 'The Hungarian method for the assignment problem'. In: *Naval research logistics quarterly* 2.1-2 (1955), pp. 83–97.

[53] Christoph Lassner, Javier Romero, Martin Kiefel, Federica Bogo, Michael J. Black and Peter V. Gehler. 'Unite the People: Closing the Loop Between 3D and 2D Human Representations'. In: *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. July 2017. URL: http://up.is.tuebingen.mpg.de.

[54] Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner. 'Gradient-based learning applied to document recognition'. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[55] Andreas M. Lehrmann, Peter V. Gehler and Sebastian Nowozin. 'A Non-parametric Bayesian Network Prior of Human Pose'. In: *The IEEE International Conference on Computer Vision (ICCV)*. Dec. 2013.

[56] Jiefeng Li, Can Wang, Hao Zhu, Yihuan Mao, Hao-Shu Fang and Cewu Lu. 'CrowdPose: Efficient Crowded Scenes Pose Estimation and A New Benchmark'. In: *arXiv preprint arXiv:1812.00324* (2018).

[57] Sijin Li and Antoni B Chan. '3d human pose estimation from monocular images with deep convolutional neural network'. In: *Asian Conference on Computer Vision*. Springer. 2014, pp. 332–347.

[58] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C Lawrence Zitnick. 'Microsoft coco: Common objects in context'. In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[59] lowerquality. *Gentle forced aligner [Computer software]*. 2020. URL: https://github.com/lowerquality/gentle (visited on 05/05/2020).

[60] Akara Robotics Ltd. *Stevie The Robot*. 2019. URL: https://stevietherobot.com/ (visited on 23/01/2020).

[61] Diogo C. Luvizon, Hedi Tabia and David Picard. 'Human pose regression by combining indirect part detection and contextual information'. In: *Computers & Graphics* 85 (Dec. 2019), pp. 15–22. ISSN: 0097-8493. DOI: 10.1016/j.cag.2019.09.002. URL: http://dx.doi.org/10.1016/j.cag.2019.09.002.

[62] Andrew L Maas, Awni Y Hannun and Andrew Y Ng. 'Rectifier nonlinearities improve neural network acoustic models'. In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[63] Timo von Marcard, Roberto Henschel, Michael Black, Bodo Rosenhahn and Gerard Pons-Moll. 'Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera'. In: *European Conference on Computer Vision (ECCV)*. Sept. 2018.

[64] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu and Xiaoqiang Zheng. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. URL: `http://tensorflow.org/`.

[65] Julieta Martinez, Rayat Hossain, Javier Romero and James J Little. 'A simple yet effective baseline for 3d human pose estimation'. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2640–2649.

[66] David McNeill. *Hand and mind: What gestures reveal about thought*. University of Chicago press, 1992.

[67] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu and Christian Theobalt. 'Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision'. In: *3D Vision (3DV), 2017 Fifth International Conference on*. IEEE. 2017. DOI: `10.1109/3dv.2017.00064`. URL: `http://gvv.mpi-inf.mpg.de/3dhp_dataset`.

[68] Dushyant Mehta, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll and Christian Theobalt. 'Single-Shot Multi-person 3D Pose Estimation from Monocular RGB'. In: *2018 International Conference on 3D Vision (3DV)* (Sept. 2018). DOI: `10.1109/3dv.2018.00024`. URL: `http://dx.doi.org/10.1109/3DV.2018.00024`.

[69] Mehdi Mirza and Simon Osindero. 'Conditional Generative Adversarial Nets'. In: (2014). arXiv: `1411.1784 [cs.LG]`.

[70] Gyeongsik Moon, Ju Yong Chang and Kyoung Mu Lee. 'Camera Distance-aware Top-down Approach for 3D Multi-person Pose Estimation from a Single RGB Image'. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2019, pp. 10133–10142.

[71] Gonçalo Mordido, Haojin Yang and Christoph Meinel. 'Dropout-GAN: Learning from a Dynamic Ensemble of Discriminators'. In: (2018). arXiv: 1807.11346 [cs.LG].

[72] Alexander Mordvintsev, Christopher Olah and Mike Tyka. *Inceptionism: Going Deeper into Neural Networks*. Google AI. 17th June 2015. URL: https://ai.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html (visited on 23/01/2020).

[73] Alexander Mordvintsev, Michael Tyka and Christopher Olah. *DeepDream*. Google AI. 2015. URL: https://github.com/google/deepdream (visited on 05/05/2020).

[74] Vaishnavh Nagarajan and J. Zico Kolter. 'Gradient descent GAN optimization is locally stable'. In: (2017). arXiv: 1706.04156 [cs.LG].

[75] Vinod Nair and Geoffrey E Hinton. 'Rectified linear units improve restricted boltzmann machines'. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*. 2010, pp. 807–814.

[76] Alejandro Newell, Kaiyu Yang and Jia Deng. 'Stacked Hourglass Networks for Human Pose Estimation'. In: *Lecture Notes in Computer Science* (2016), pp. 483–499. ISSN: 1611-3349. DOI: 10.1007/978-3-319-46484-8_29. URL: http://dx.doi.org/10.1007/978-3-319-46484-8_29.

[77] Aiden Nibali, Zhen He, Stuart Morgan and Luke Prendergast. '3D Human Pose Estimation With 2D Marginal Heatmaps'. In: *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)* (Jan. 2019). DOI: 10.1109/wacv.2019.00162. URL: http://dx.doi.org/10.1109/wacv.2019.00162.

[78] Aiden Nibali, Zhen He, Stuart Morgan and Luke Prendergast. 'Numerical Coordinate Regression with Convolutional Neural Networks'. In: (2018). arXiv: 1801.07372 [cs.CV].

[79] Guanghan Ning and Heng Huang. 'LightTrack: A Generic Framework for Online Top-Down Human Pose Tracking'. In: (2019). arXiv: 1905.02822 [cs.CV].

[80] Chris Olah, Alexander Mordvintsev and Ludwig Schubert. 'Feature Visualization'. In: *Distill* (2017). https://distill.pub/2017/feature-visualization. DOI: 10.23915/distill.00007.

[81] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai and

Soumith Chintala. 'PyTorch: An Imperative Style, High-Performance Deep Learning Library'. In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[82] Dario Pavllo, Christoph Feichtenhofer, David Grangier and Michael Auli. '3D human pose estimation in video with temporal convolutions and semi-supervised training'. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[83] Stanislav Pidhorskyi, Donald Adjeroh and Gianfranco Doretto. 'Adversarial Latent Autoencoders'. In: (2020). arXiv: 2004.04467 [cs.LG].

[84] Alec Radford, Luke Metz and Soumith Chintala. 'Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks'. In: (2015). arXiv: 1511.06434 [cs.LG].

[85] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei and Ilya Sutskever. 'Language models are unsupervised multitask learners'. In: *OpenAI Blog* 1.8 (2019), p. 9.

[86] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun. 'Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (June 2017), pp. 1137–1149. ISSN: 2160-9292. DOI: 10.1109/tpami.2016.2577031. URL: http://dx.doi.org/10.1109/TPAMI.2016.2577031.

[87] Helge Rhodin, Mathieu Salzmann and Pascal Fua. 'Unsupervised Geometry-Aware Representation for 3D Human Pose Estimation'. In: *Lecture Notes in Computer Science* (2018), pp. 765–782. ISSN: 1611-3349. DOI: 10.1007/978-3-030-01249-6_46. URL: http://dx.doi.org/10.1007/978-3-030-01249-6_46.

[88] Matteo Ruggero Ronchi and Pietro Perona. 'Benchmarking and Error Diagnosis in Multi-instance Pose Estimation'. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (Oct. 2017). DOI: 10.1109/iccv.2017.48. URL: http://dx.doi.org/10.1109/ICCV.2017.48.

[89] Olaf Ronneberger, Philipp Fischer and Thomas Brox. 'U-Net: Convolutional Networks for Biomedical Image Segmentation'. In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (2015), pp. 234–241. ISSN: 1611-3349. DOI: 10.1007/978-3-319-24574-4_28. URL: http://dx.doi.org/10.1007/978-3-319-24574-4_28.

[90] Frank Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

[91] David E Rumelhart, Geoffrey E Hinton and Ronald J Williams. 'Learning representations by back-propagating errors'. In: *nature* 323.6088 (1986), pp. 533–536.

[92] Marta Sanzari, Valsamis Ntouskos and Fiora Pirri. 'Bayesian Image Based 3D Pose Estimation'. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe and Max Welling. Cham: Springer International Publishing, 2016, pp. 566–582. ISBN: 978-3-319-46484-8.

[93] Jürgen Schmidhuber. 'Generative Adversarial Networks are special cases of Artificial Curiosity (1990) and also closely related to Predictability Minimization (1991)'. In: *Neural Networks* 127 (July 2020), pp. 58–66. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2020.04.008. URL: http://dx.doi.org/10.1016/j.neunet.2020.04.008.

[94] Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert and Zehan Wang. 'Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016). DOI: 10.1109/cvpr.2016.207. URL: http://dx.doi.org/10.1109/CVPR.2016.207.

[95] L. Sigal and M. J. Black. 'Measure Locally, Reason Globally: Occlusion-sensitive Articulated Pose Estimation'. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*. Vol. 2. June 2006, pp. 2041–2048. DOI: 10.1109/CVPR.2006.180.

[96] Leonid Sigal, Alexandru O Balan and Michael J Black. 'Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion'. In: *International journal of computer vision* 87.1-2 (2010), p. 4.

[97] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton et al. 'Mastering the game of go without human knowledge'. In: *Nature* 550.7676 (2017), pp. 354–359.

[98] Tomas Simon, Hanbyul Joo, Iain Matthews and Yaser Sheikh. 'Hand Keypoint Detection in Single Images using Multiview Bootstrapping'. In: *CVPR*. 2017.

[99] Karen Simonyan, Andrea Vedaldi and Andrew Zisserman. 'Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps'. In: (2013). arXiv: 1312.6034 [cs.CV].

[100] Karen Simonyan and Andrew Zisserman. 'Very deep convolutional networks for large-scale image recognition'. In: *arXiv preprint arXiv:1409.1556* (2014).

[101] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. 'Dropout: A Simple Way to Prevent Neural Networks from Overfitting'. In: *J. Mach. Learn. Res.* 15.1 (Jan. 2014), pp. 1929–1958. ISSN: 1532-4435.

[102] Ke Sun, Bin Xiao, Dong Liu and Jingdong Wang. 'Deep High-Resolution Representation Learning for Human Pose Estimation'. In: *CVPR*. 2019.

[103] Xiao Sun, Chuankang Li and Stephen Lin. 'Explicit Pose Deformation Learning for Tracking Human Poses'. In: *arXiv preprint arXiv:1811.07123* (2018).

[104] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang and Yichen Wei. 'Integral human pose regression'. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 529–545.

[105] Ilya Sutskever, Oriol Vinyals and Quoc V Le. 'Sequence to sequence learning with neural networks'. In: *Advances in neural information processing systems*. 2014, pp. 3104–3112.

[106] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. 'Going Deeper with Convolutions'. In: (2014). arXiv: 1409.4842 [cs.CV].

[107] Mingxing Tan and Quoc V Le. 'Efficientnet: Rethinking model scaling for convolutional neural networks'. In: *arXiv preprint arXiv:1905.11946* (2019).

[108] T. Tieleman and G. Hinton. *Lecture 6.5—RMSprop: Divide the gradient by a running average of its recent magnitude*. COURSERA: Neural Networks for Machine Learning. 2012.

[109] Alexander Toshev and Christian Szegedy. 'Deeppose: Human pose estimation via deep neural networks'. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1653–1660.

[110] Matt Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton and John Collomosse. 'Total Capture: 3D Human Pose Estimation Fusing Video and Inertial Sensors'. In: *2017 British Machine Vision Conference (BMVC)*. 2017.

[111] Dmitry Ulyanov, Andrea Vedaldi and Victor Lempitsky. 'It Takes (Only) Two: Adversarial Generator-Encoder Networks'. In: (2017). arXiv: 1704.02304 [cs.CV].

[112] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev and Cordelia Schmid. 'Learning from Synthetic Humans'. In: *CVPR*. 2017.

[113] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. 'Attention Is All You Need'. In: (2017). arXiv: 1706.03762 [cs.CL].

[114] Andrey Voynov and Artem Babenko. 'Unsupervised Discovery of Interpretable Directions in the GAN Latent Space'. In: (2020). arXiv: 2002.03754 [cs.LG].

[115] Keze Wang, Liang Lin, Chenhan Jiang, Chen Qian and Pengxu Wei. '3D Human Pose Machines with Self-supervised Learning'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019), pp. 1–1. ISSN: 1939-3539. DOI: 10.1109/tpami.2019.2892452. URL: http://dx.doi.org/10.1109/TPAMI.2019.2892452.

[116] Shih-En Wei, Varun Ramakrishna, Takeo Kanade and Yaser Sheikh. 'Convolutional Pose Machines'. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016). DOI: 10.1109/cvpr.2016.511. URL: http://dx.doi.org/10.1109/CVPR.2016.511.

[117] Sabine Wieluch and Friedhelm Schwenker. 'Dropout Induced Noise for Co-Creative GAN Systems'. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (Oct. 2019). DOI: 10.1109/iccvw.2019.00383. URL: http://dx.doi.org/10.1109/ICCVW.2019.00383.

[118] Pieter Wolfert, Taras Kucherenko, H Kjelström and Tony Belpaeme. 'Should beat gestures be learned or designed? A benchmarking user study'. In: *ICDL-EPIROB 2019 Workshop on Naturalistic Non-Verbal and Affective Human-Robot Interactions*. 2019, pp. 1–4.

[119] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo and Ross Girshick. *Detectron2. Detectron2 is FAIR's next-generation platform for object detection and segmentation*. 2019. URL: https://github.com/facebookresearch/detectron2.

[120] Donglai Xiang, Hanbyul Joo and Yaser Sheikh. 'Monocular total capture: Posing face, body, and hands in the wild'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 10965–10974.

[121] Bin Xiao, Haiping Wu and Yichen Wei. 'Simple Baselines for Human Pose Estimation and Tracking'. In: *Lecture Notes in Computer Science* (2018), pp. 472–487. ISSN: 1611-3349. DOI: 10.1007/978-3-030-01231-1_29. URL: http://dx.doi.org/10.1007/978-3-030-01231-1_29.

[122] Yuliang Xiu, Jiefeng Li, Haoyu Wang, Yinghong Fang and Cewu Lu. 'Pose Flow: Efficient Online Pose Tracking'. In: *BMVC*. 2018.

[123] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li and Xiaogang Wang. '3d human pose estimation in the wild by adversarial learning'. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 5255–5264.

[124] Youngwoo Yoon, Woo-Ri Ko, Minsu Jang, Jaeyeon Lee, Jaehong Kim and Geehyuk Lee. 'Robots learn social skills: End-to-end learning of co-speech gesture generation for humanoid robots'. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 4303–4309.

[125] youtube-dl developers. *youtube-dl [Computer software]*. 2020. URL: `https://github.com/ytdl-org/youtube-dl` (visited on 05/05/2020).

[126] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue and Yichen Wei. *Code repository for Towards 3D Human Pose Estimation in the Wild: a Weakly-supervised Approach*. 2017. URL: `https://github.com/xingyizhou/pose-hg-3d` (visited on 08/05/2020). 'Towards 3D Human Pose Estimation in the Wild: A Weakly-Supervised Approach'. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[127] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue and Yichen Wei. 'Towards 3D Human Pose Estimation in the Wild: A Weakly-Supervised Approach'. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.