

# FASHION AND APPAREL CLASSIFICATION USING CONVOLUTIONAL NEURAL NETWORKS

Alexander Schindler

[alexander.schindler@ait.ac.at](mailto:alexander.schindler@ait.ac.at)

Thomas Lidy

[lidy@ifs.tuwien.ac.at](mailto:lidy@ifs.tuwien.ac.at)

Stephan Karner

[stephan.karner@monstyle.io](mailto:stephan.karner@monstyle.io)

Matthias Hecker

[matthias.hecker@monstyle.io](mailto:matthias.hecker@monstyle.io)

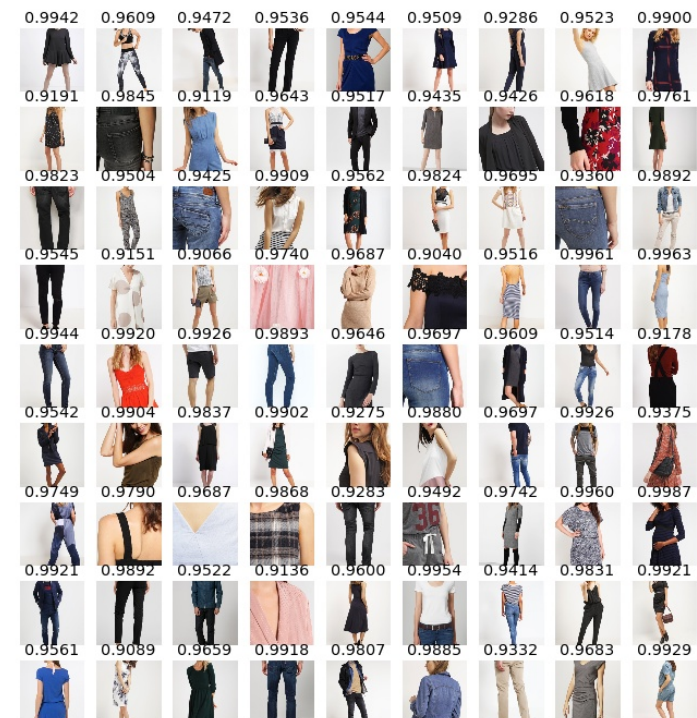


# FASHION IMAGE CLASSIFICATION

- Online e-commerce access to product images
  - Asos-EU, Farfetch, Zalando
    - Images & metadat
- **Problem**
  - Metadata differs in
    - Quality, granularity, taxonomy
    - Taxonomy varies in depth of categorical hirarchy
- **Task**
  - use CNNs to
    - Consolidate Metadta
    - Enrich Metadata

# BRIEF OVERVIEW

- Empirical study
- Applying deep Convolutional Neural Networks to fashion classification
- Evaluated five CNN architectures
- Custom and pre-trained models
- Evaluated on three tasks
  - Person detection
  - Product classification
  - Gender prediction



# DATASETS

## • Person

- 7.833 images
  - 5.669 labeled as persons
  - 2.164 labeled as products

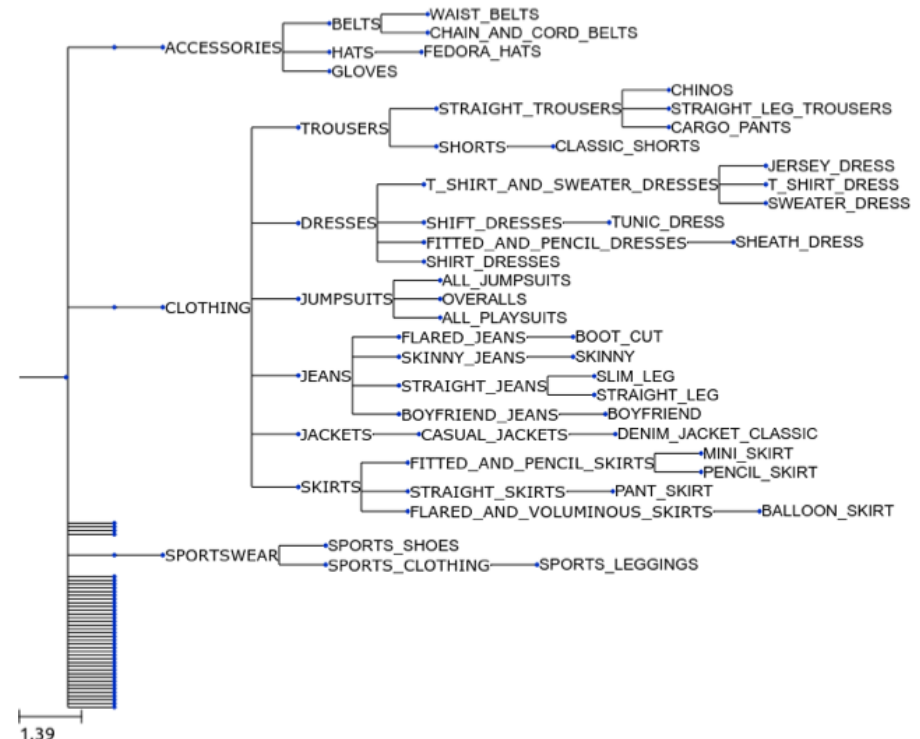
## • Products

- 234.884 images
- 39.474 products
  - ~5,95 images per product
- Ground-truth labels assignments
  - Product category
    - Label hierarchy
  - Gender
  - Age

Person



Product



# DATA QUALITY / ISSUES

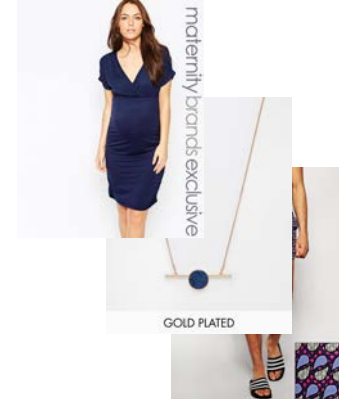
White background



Worn by persons



Text, Overlays



Close-up texture



Close-up fit



Multiple objects



Brand logo

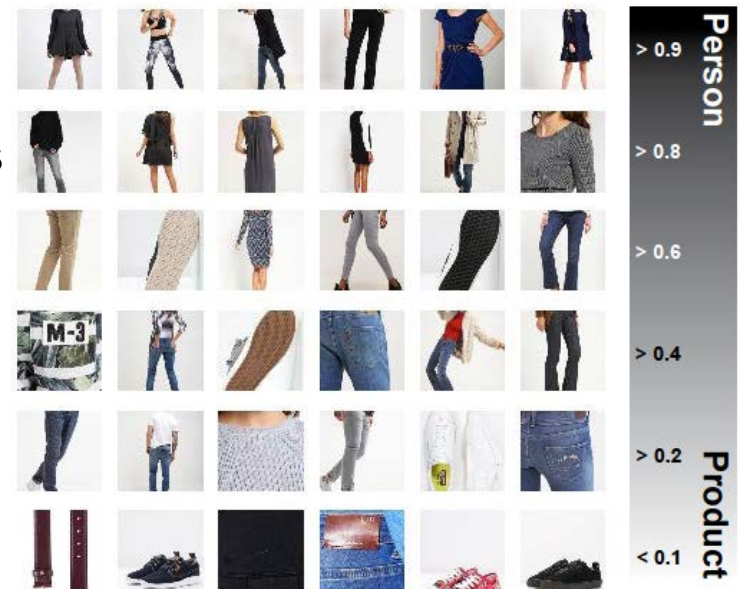


Misc



# PERSON DETECTION

- **Products also presented by persons**
  - How they look when worn?
- **Problem**
  - Person wears multiple products
  - Single-label classification
  - Decision problem
- **Approach**
  - Train model to identify persons
  - Use model to filter images with persons
  - VGG-like custom model
- **Results**
  - **91.07%** accuracy on persons dataset



# PRODUCT CLASSIFICATION

- **Deep Neural Network Architectures**

- Vgg16 and Vgg19
- InceptionV3
- Custom CNN and Vgg-like

- **Experiments**

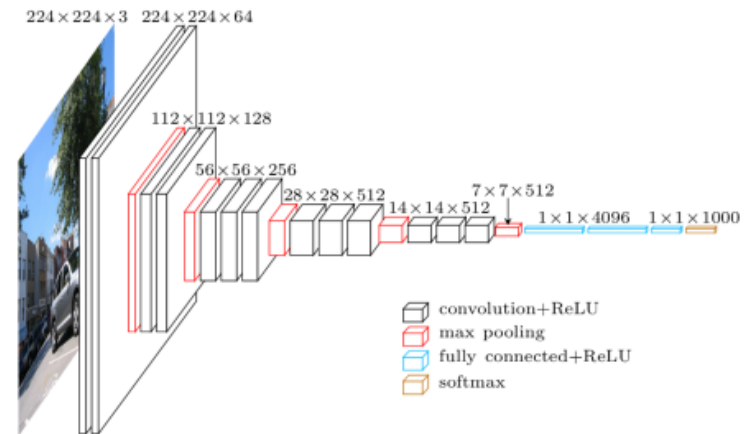
- From-Scratch
- Pre-Trained

- **Evaluation**

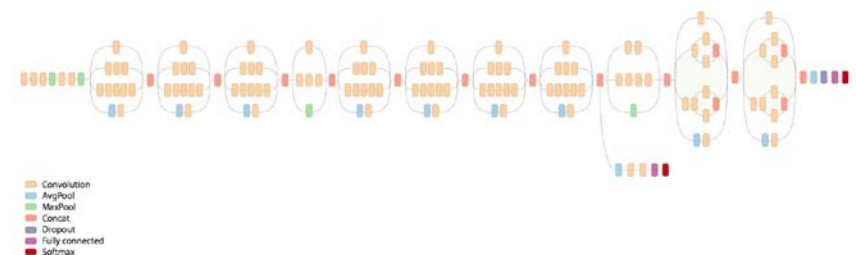
- 4-Fold Crossvalidation
- Grouped Stratification

- **Metrics**

- Raw Accuracy
- Max of Sum per product



David Fossard, <https://www.cs.toronto.edu/~frossard/post/vgg16/>



John Shlens, <https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html>

# EXPERIMENTAL SET-UP

- **Small scale**
  - Subset of 23.305 images
- **Large scale**
  - 234.408 images
- **All Models**
  - Data Augmentation
    - 25% vertically and horizontally shifting
    - 25% zoom range
    - Horizontal flipping



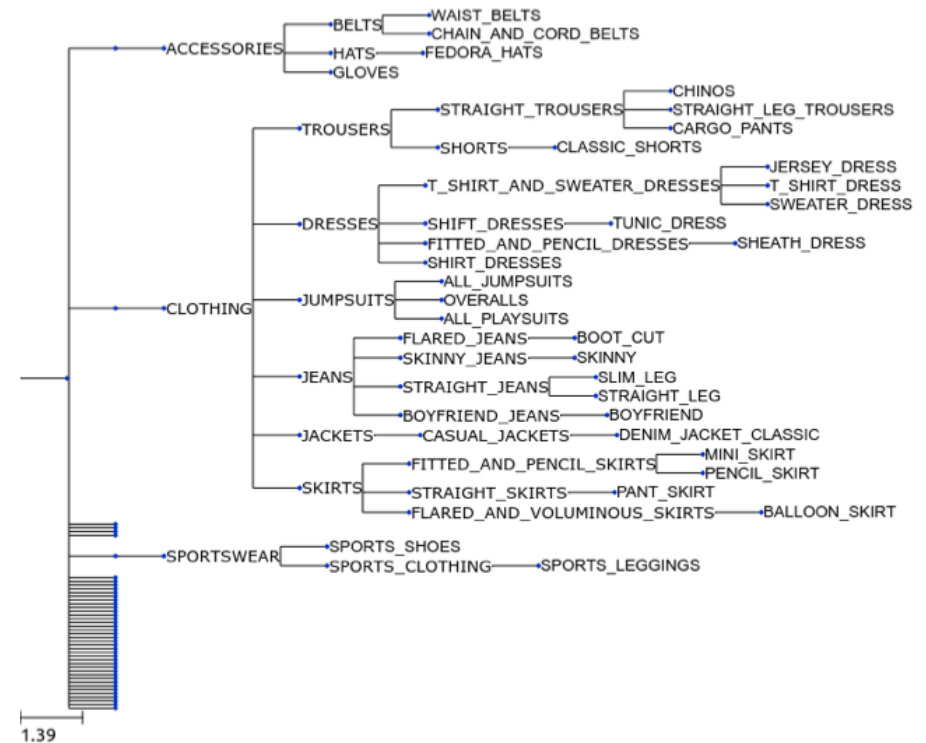
## RESULTS – SMALL SCALE (24K)

- **Best results:** Pre-trained + fine-tune entire model
  - Freezing network + training only top layers not as good
- **Person filter** did not improve performance
- Small custom models have advantage of speed, but not as accurate

Description	best fold	best fold cum max	Mean cum max
InceptionV3, pretrained, fine-tuned	0.706	0.794	0.791
InceptionV3, pretrained, fine-tuned	0.658	0.729	0.716
VGG16, pretrained, fine-tuned	0.646	0.711	0.691
InceptionV3, pretrained, fine-tuned, person filter model as layer	0.569	0.685	0.658
VGG19, pretrained, fine-tuned	0.579	0.673	0.634
InceptionV3, pretrained, fine-tuned, no augmentation	0.564	0.673	0.647
VGG19, pretrained, train only top-layers	0.578	0.669	0.343
VGG16, pretrained, train only top-layers	0.603	0.652	0.368
InceptionV3, pretrained, train only top-layers	0.585	0.650	0.643
InceptionV3, pretrained, fine-tuned - person filtered metadata	0.640	0.636	0.614
InceptionV3, clean	0.492	0.594	0.580
Custom CNN, augmentation	0.506	0.568	0.538
Custom CNN	0.463	0.556	0.523
Custom VGG-like	0.438	0.549	0.519
VGG16, clean	0.439	0.455	0.443
VGG19, clean	0.437	0.447	0.430
VGG19, pretrained, train only top-layers	0.819	0.887	0.880
InceptionV3, pretrained, fine-tuned	0.798	0.863	0.836
VGG19, pretrained, fine-tuned	0.762	0.846	0.830

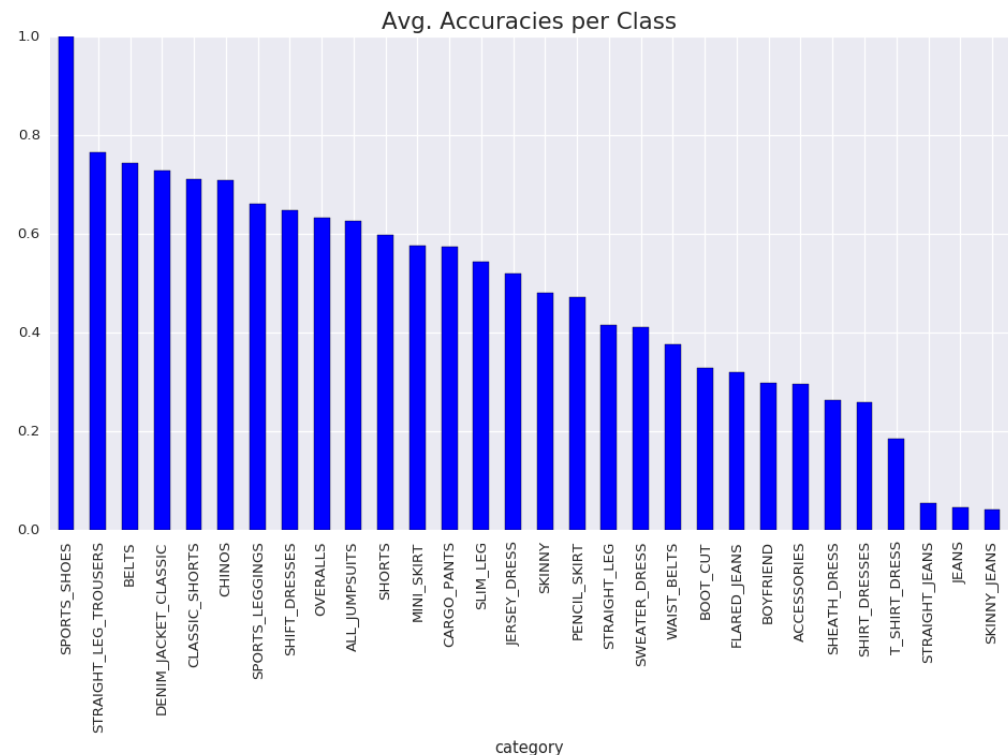
# CONFUSIONS – SMALL SCALE (24K)

		Confusion matrix																															
True label	CHINOS	35	2	23	0	0	0	0	4	0	0	0	4	3	0	4	1	11	0	0	0	1	24	0	0	0	3	1	13				
	CLASSIC_SHORTS	1	47	39	11	4	14	11	42	173	67	14	1	611	51	75	19	1211	289	89	20	33	0	63	15	24	0	14	6	57	6	0	
	ALL_JUMPSUITS	7	7	2335	0	0	0	1	2	1	0	0	14	2	0	0	2	121	2	1	0	0	2	1	608	0	1	0	1	0	36		
	JERSEY_DRESS	0	6	3	566	62	1	9	4	9	236	4	1	4	7	0	0	2	0	18	99	3	278	0	0	10	398	2	0	0	0		
	SPORTS_SHOES	0	21	2	44	951	16	135	20	19	25	13	1	10	42	2	0	5	1	92	123	6	956	2	6	27	670	3	1	1	0		
	SHIFT_DRESSES	0	7	1	0	7	479	89	146	3	0	1	3	0	7	0	0	13	1	8	7	1	46	0	1	0	11	2	0	1	0		
	BELTS	1	27	1	4	97	48	31	42	254	14	7	67	8	6	9	3	0	13	4	18	49	0	404	4	8	2	111	133	1	0	1	
	OVERALLS	2	204	6	1	14	139	288	46	12	9	1	2	141	182	45	36	6	210	18	365	24	0	50	23	34	0	22	30	17	3	0	
	SKINNY	0	40	1	5	6	8	17	18	933	7	0	41	12	20	4	0	45	5	9	19	1	50	0	1	0	28	3	5	2	0	0	
	SLIM_LEG	0	8	1	298	29	0	1	4	9	311	2	2	2	6	0	0	4	0	16	73	8	103	1	0	3	93	0	1	0	0	0	
	STRAIGHT_LEG	0	1	2	1	24	1	68	8	1	3	18	2	0	1	1	0	0	1	1	43	5	143	3	0	8	51	2	1	0	0	0	
	JEANS	4	496	9	1	6	0	5	84	36	1	0	6880	106	23	79	626	387	2110	9	14	0	24	23	47	1	2	5	764	8	2	0	
	SHIRT_DRESSES	5	50	3	1	9	0	11	164	9	1	0	2211804	12	235	27	382	15	20	10	0	26	16	24	0	4	3	77	0	0	0	0	
	CARGO_PANTS	0	88	0	2	25	4	6	34	20	4	0	34	7	898	1	0	59	5	19	23	0	48	2	5	0	17	1	1	1	1	1	
	SHEATH_DRESS	1	34	3	0	5	1	11	47	7	2	0	214368	2	982	50	209	6	7	15	0	10	6	13	0	2	7	76	1	1	1	1	
	STRAIGHT_LEG_TROUSERS	0	108	1	0	1	0	1	13	2	0	0	927	33	2	42	8471044	19	0	1	0	2	2	4	0	0	0	174	0	3	3	0	
	SKINNY_JEANS	2	998	17	0	4	1	5	138	64	4	0	3531269	45	92	900378382	9	10	0	19	13	76	0	4	3	721	8	3	21	8	0	0	
	BOYFRIEND	1	176	1	0	1	3	1	20	23	1	0	242	24	4	9	21	1070581	3	0	0	8	0	15	0	1	0	40	0	0	0	0	0
	MINI_SKIRT	0	20	3	11	52	9	20	474	15	2	3	14	19	28	4	0	15	4	174893	3	247	2	1	5	132	0	5	0	0	0	0	0
	SPORTS_LEGGINGS	0	13	1	97	102	5	27	18	32	25	20	11	8	15	4	1	10	1	883127	72	2451	14	12	33	307	1	5	2	0	0	0	0
PENCIL_SKIRT	0	0	0	11	11	0	0	0	2	1	8	1	0	0	0	0	0	2	149	19	200	0	1	11	36	0	0	0	0	0	0	0	
SHORTS	0	25	10	182	694	22	294	41	55	49	71	22	16	50	7	5	10	3	1982292	687411	9	16	1761882	5	3	3	0	3	0	0	0	0	
DENIM_JACKET_CLASSIC	1	7	0	0	1	1	5	31	0	0	1	22	0	0	6	2	9	1	3	37	0	51	894	16	0	1	1	6	0	0	0	0	
SWEATER_DRESS	5	4	0	0	1	1	2	4	0	1	0	12	4	1	1	1	14	0	3	1	0	2	1	400	0	1	0	5	0	0	0	0	
BOOT_CUT	0	0	1	9	28	0	8	2	5	1	11	0	0	0	2	0	1	0	5	65	9	292	0	0	36	182	1	0	0	0	0	0	
STRAIGHT_JEANS	0	9	3	246	530	7	131	14	24	64	40	8	8	19	0	0	3	0	85	253	201819	0	9	852401	7	2	1	0	0	0	0	0	
FLARED_JEANS	0	7	0	1	0	1	108	17	3	0	5	2	2	0	0	1	13	1	1	0	0	20	1	0	0	6	618	1	0	0	0	0	
ACCESSORIES	0	94	1	0	1	1	4	19	8	0	0	122359	8	32	90	829	19	3	5	0	7	4	15	0	4	1	1704	3	2	2	0	0	
WAIST_BELTS	0	7	0	0	1	0	0	1	4	0	0	6	1	0	0	0	12	0	0	2	1	3	0	2	0	1	0	7	11	0	0	0	
T_SHIRT_DRESS	2	2	2	65	0	0	0	0	1	0	0	11	5	0	1	2	6	1	0	3	1	3	2	11	0	0	0	2	0	71	0	0	
		CHINOS	CLASSIC_SHORTS	ALL_JUMPSUITS	JERSEY_DRESS	SPORTS_SHOES	SHIFT_DRESSES	BELTS	OVERALLS	SKINNY	SLIM_LEG	STRAIGHT_LEG	JEANS	SHIRT_DRESSES	CARGO_PANTS	CHEATO_PANTS	STRAIGHT_LEG_TROUSERS	SKINNY_JEANS	BOYFRIEND	MINI_SKIRT	SPORTS_LEGGINGS	PENCIL_SKIRT	SHORTS	DENIM_JACKET_CLASSIC	SWEATER_DRESS	BOOT_CUT	STRAIGHT_JEANS	FLARED_JEANS	ACCESSORIES	WAIST_BELTS	T_SHIRT_DRESS		



# PER CLASS ACCURACIES – LARGE SCALE (234K)

- Problem of different granularity of provided ground truth
- Parent/Child nodes used interchangeably
  - Misclassification of child as parent is not wrong
  - Model does not consider hierarchy



# CONCLUSIONS

- Despite large dataset and reduced number of classes
  - **Pretrained models outperform** from-scratch training
    - Product classification – 79.1%
    - Gender prediction – 88.0%
- **Custom small model** enough to learn **binary task**
  - person/product classification – 91.07%
- **Preprocessing** of **ground-truth** required
  - Flatten hierarchy, remove ambiguities and overlaps
  - Use **hierarchical CNNs**
  - Use **attention** (person images)

# THANK YOU!

Alexander Schindler, 08.06.2018



# Machine Learning Showcase

Mon Style GmbH

<https://monstyle.io>

<https://asksophie.io>

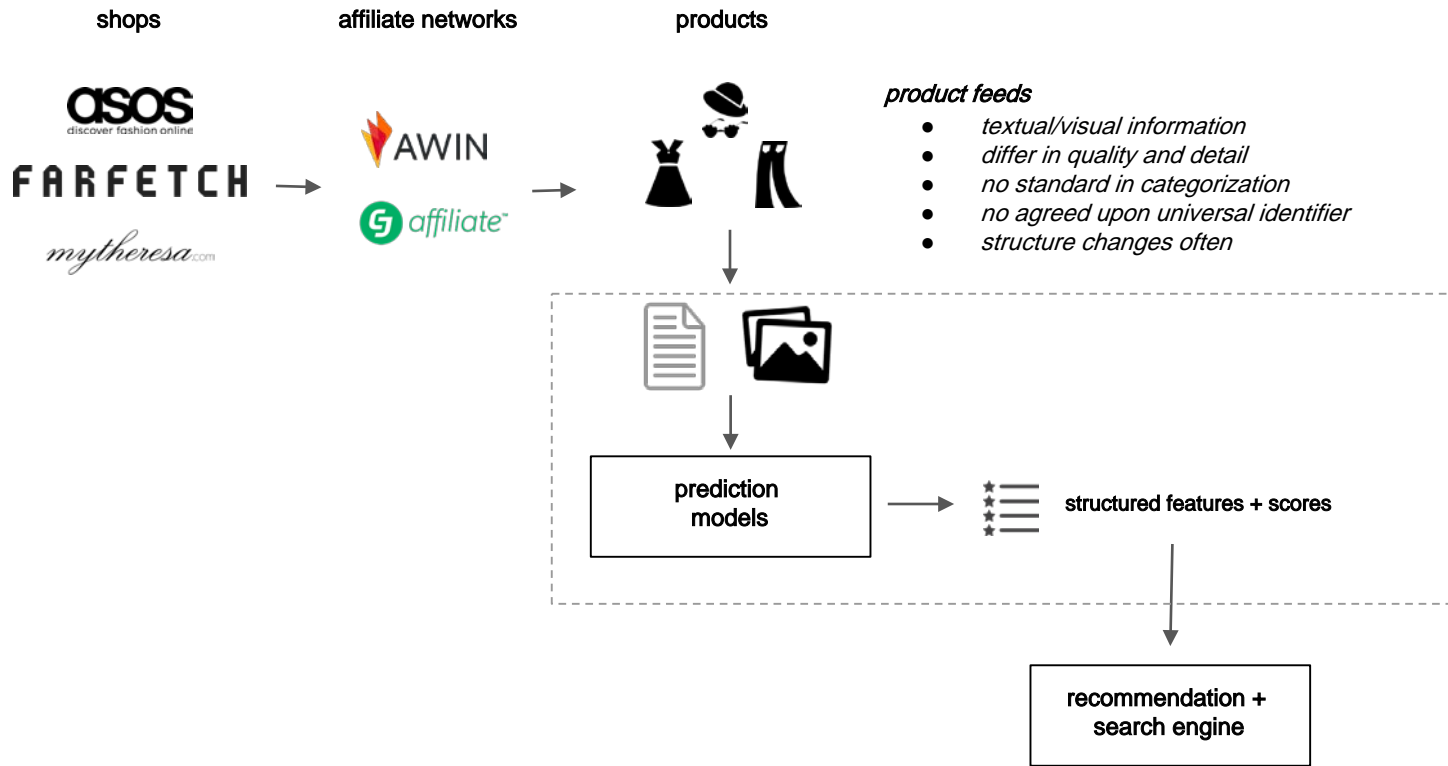


# Presentation Agenda

- 1. Use Case / Problem Description  
(what are we talking about)
- 1. Prediction Pipeline  
(product data to curated catalog)
- 1. Datasets with Mechanical Turk  
(example dataset created with mturk)

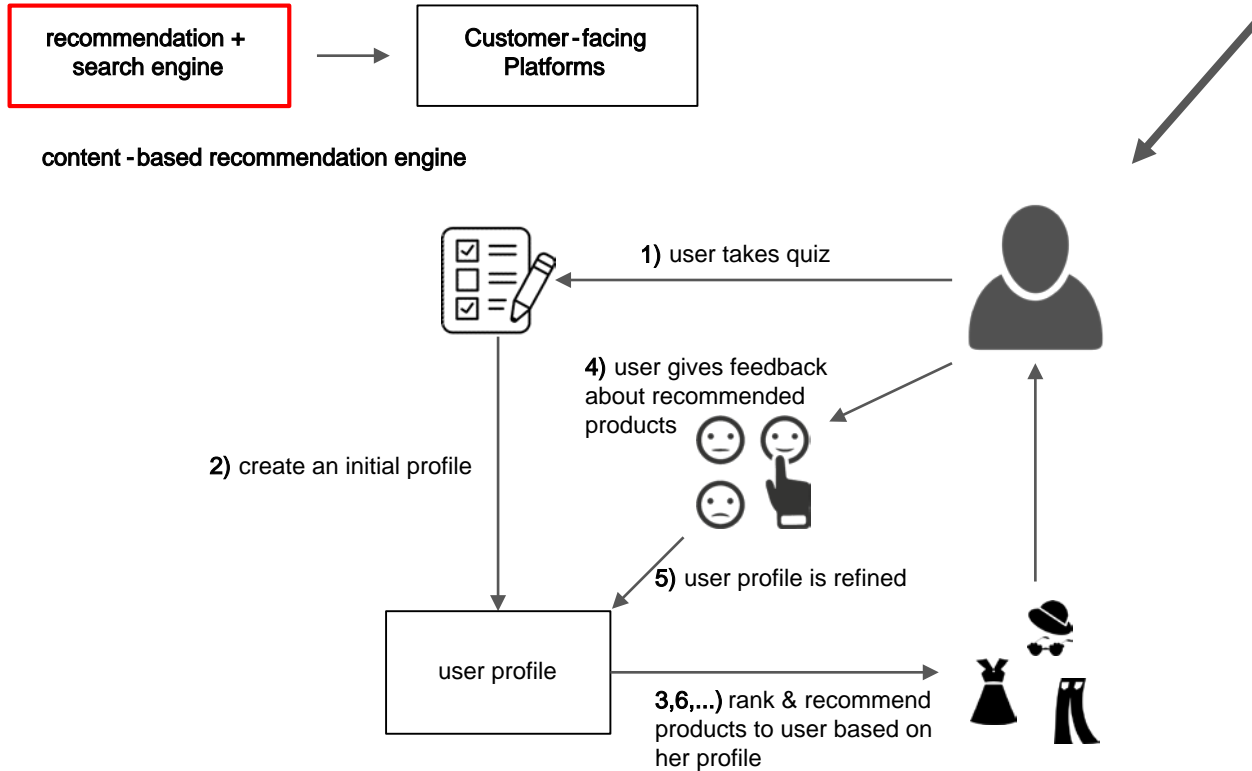


# 1. Affiliate Overview





# 1. Recommendation



# 1. Website

recommendation +  
search engine



Customer-facing  
Platforms

web shopping platform & community



↪ [monstyle.io](https://monstyle.io)

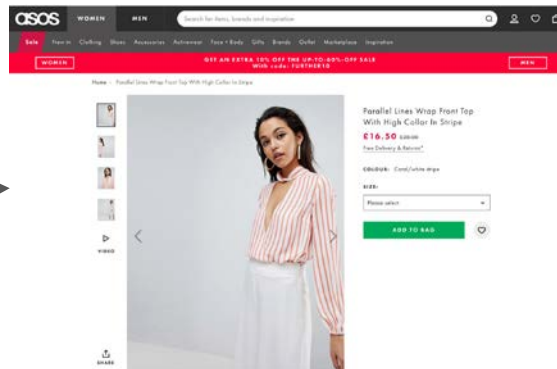
Parallel Lines

Wrap Front Top With High Collar In  
Stripe - Coral/white stripe

£16.5



Top by Parallel Lines, Choker neck,  
Saves you buying one, Cross-over  
front, Keep it under wraps, Long  
sleeves, Regular fit - true to size.  
Parallel Lines takes you from desk  
to date night with attitude. Work its  
clean and pared-back pieces 9-9 or  
boss the weekend in its wrap maxi



↪ [asos.com](https://www.asos.com)



# 1. Chatbot

recommendation +  
search engine



Customer-facing  
Platforms

facebook messenger chatbot

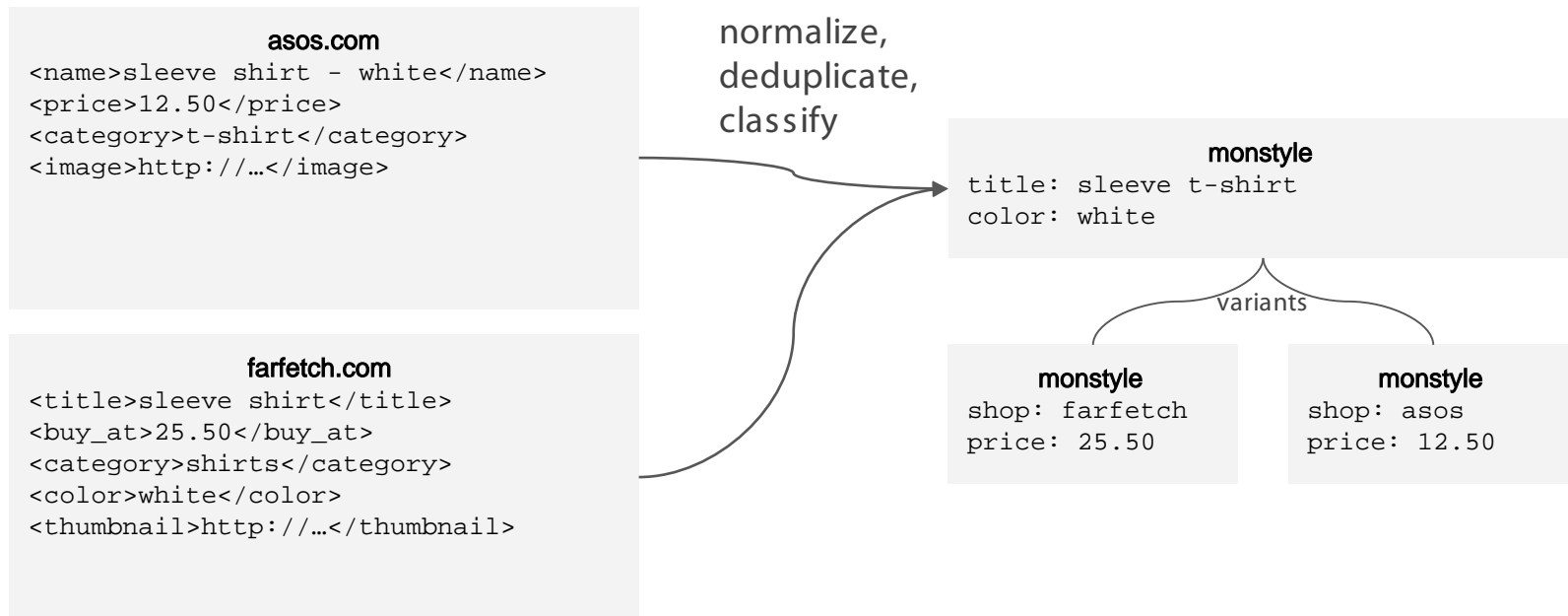
↪ messenger.com, @SophieStyleBot

The screenshot shows a chatbot conversation on Facebook Messenger. At the top, a grey bubble from the bot says: "I am your very own Personal Stylist and here to help you with your future shopping dilemmas! 😊". A timestamp "10:05PM" follows. A blue bubble from the user says: "I'm looking for a red floral dress". The bot responds with a grey bubble: "Here are results for Dress, Red, Floral Prints:". Below this is a product card for an "Oasis" dress. The card features a photo of a woman in a red floral dress, the brand name "Oasis", the description "Fit and Flare Dress | £46 | Floral Print RucheD Front Tea Dress - Multi red monstyle.io", and buttons for "Shop" and "Share". To the right of the main card is a smaller, partially visible card for an "ASOS" dress. At the bottom, a grey bubble from the bot says: "Hopefully, you are happy with the recommendations I gave you!". Below that, a blue bubble from the user asks: "More Dress, Red, Floral Prints?". The chat interface includes a profile picture of a woman, a search bar, and a bottom navigation bar with icons for home, search, and a plus sign.



# 1. Example Shop Data

data feeds provided as XML/CSV files:



# 1. Example Problem: Brand Names

some brand names include category descriptions, for instance:



**Product Title:**

Versace **Jeans** Foil Logo Tank Dress with Cutout Back

-> generate list of brand names

-> use list to remove brands from product title

**Product Title:** Foil Logo Tank Dress with Cutout Back

**Product Brand:** Versace Jeans

-> free text search engine: ranking by field, rank title matches higher than brand



# 1. Example Problem: Brand Names

## ERGEBNISSE FÜR "RED DRESS"

Ansehen: 60 180

Sortieren Unsere Auswahl ▼

1 of 14

Gefiltert nach: (2389 gefundene Artikel)

Die von Ihnen ausgewählten Filter erscheinen hier.

Kategorie

- ▶ Accessoires
- ▶ Kleidung
- ▶ Lifestyle
- ▶ Schmuck
- ▶ Schuhe
- ▶ Taschen



**RED VALENTINO**  
Red Valentino Jeans-



**RED VALENTINO**  
fitted dress



**RED VALENTINO**  
scallop dress



**RED VALENTINO**  
flared dress



# 1. Tags/Keyword Vocabulary

different terminology used around shops, needs normalization:

- > extensive list of synonyms/aliases in order to normalize data
- > keep synonyms for user search

**For Example: Lingerie and Nightwear »  
Sleepwear » Robe**

kimono robe, robe, night kimono, nightgown,  
dressing gown, night robe, peignoir, sleeping  
gown, bath robe, bathing robe

**For Example: Floral Print**

floral, leaf, floral print, leaf print



# 1. Product Catalog

for example:

<b>id</b>	there exists no universal standard, ASIN, EAN, UPC, etc.
<b>title</b>	normalized version can be generated from brand, categories and other attributes
<b>brand</b>	brands are an important indicator for style, size/fit -differences, etc.
<b>price</b>	including currency
<b>availability</b>	to not show unavailable products
<b>category</b>	hierarchical, fashion-domain specific
<b>color</b>	normalized color labels
<b>pattern</b>	floral print, ...
<b>material</b>	leather, fur, cotton, vegan?
<b>attributes</b>	some attributes are category-specific, such as collar type or skirt length





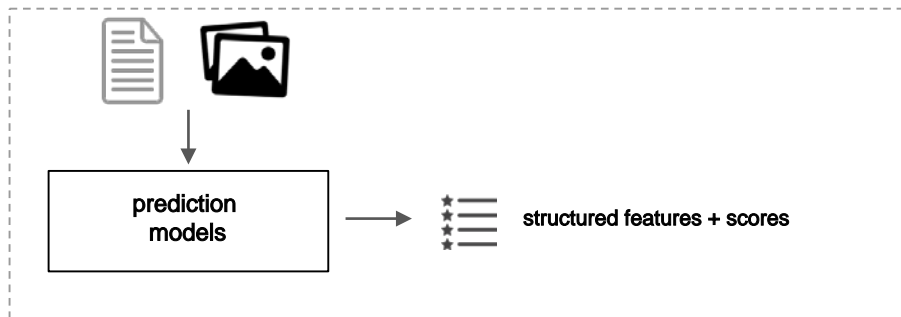
# 1. Product Catalog Example



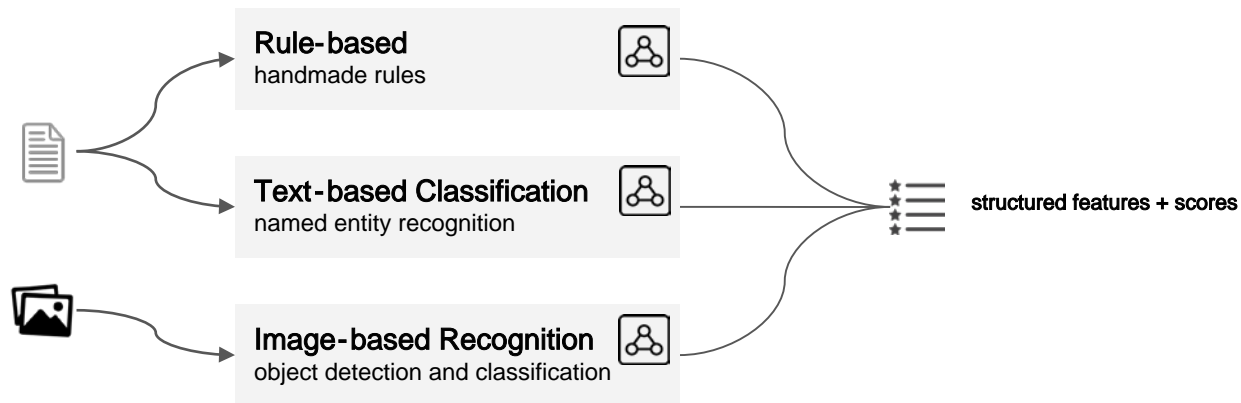
id	2791782 - 7049 - 4784 - be6b - a2eb1865994e
title	leaf-print flared midi dress
brand	OSCAR DE LA RENTA
price	2.621 €
availability	available
category	clothing » dress » fit and flare dress
color	white, blue » navy
pattern	leaf print
material	cotton
attributes	sleeveless, square angular neckline, kneelength, ...



## 2. Prediction Pipeline



## 2. Prediction Models



## 2. Example Input

### For Example

```
<name>leaf-print flared midi dress</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>
```



#### Rule-based

handmade rules



#### Text-based Classification

named entity recognition



#### Image-based Recognition

object detection and classification



# 2. Rule Model

## Rule-based Recognition

```
<name>leaf-print flared midi dress</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>
```



Rule-based  
handmade rules



Remove Rule

raw.buy\_price

Type Equality

float

Dynamic Copy

value

Remove Match

Create Rule

Clone Match

Ruleset Management UI

```
return raw.buy_price
```

Price: 2,621.00  
→ 100.0



# 2. Rule Model

## Rule-based Recognition

```
<name>leaf-print flared midi dress</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>
```



Rule-based  
handmade rules



Remove Rule

raw.gender

?

Equality

female

Static Set

category

Female

x

Remove Match

Create Rule

Clone Match

Ruleset Management UI

```
if raw.gender=='female':  
    return GENDER.FEMALE
```



**Gender:** female  
→ 100.0



# 2. NLP Model

## Named Entity Recognition

```
<name>leaf-print flared midi dress</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>
```



**Text-based Classification**  
named entity recognition



leaf-print flared midi dress

Navy blue and white cotton-blend leaf-print

flared midi dress from Oscar de la Renta.

**Category:** clothing » dress  
» fit and flare dress → 2.0

**Print:** floral print → 2.0

**Color:** white → 1.0

**Color:** blue » navy blue  
→ 1.0

**Material:** natural » cotton  
→ 1.0



# 2. Vision Model

## Image Classification

`<name>leaf-print flared midi dress</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>`



**Image-based Recognition**  
object detection and classification



**Gender:** female  
→ 0.9289139

**Category:** clothing » dress  
→ 0.9762532





## 2. Feature Assembling

### Assemble Final Predicted Features



## 2. Example: Incorrect Input

### Wrong Data

title wrong

<name>leaf-print skirt</name>  
<buy\_price>2621</buy\_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>



#### Rule-based

handmade rules



#### Text-based Classification

named entity recognition



#### Image-based Recognition

object detection and classification



## 2. Example: Incorrect Input

### Named Entity Recognition

```
<name>leaf-print skirt</name>  
<buy_price>2621</buy_price>  
<gender>female</gender>  
<description>Navy blue and white cotton-blend  
leaf-print flared midi dress from Oscar de la  
Renta.</description>
```



**Text-based Classification**  
named entity recognition



leaf-print skirt

Navy blue and white cotton-blend leaf-print

flared midi dress from Oscar de la Renta.

**Category:** clothing » skirt  
→ 1.0

**Category:** clothing » dress  
» fit and flare dress → 1.0

**Print:** floral print → 2.0

**Color:** white → 1.0

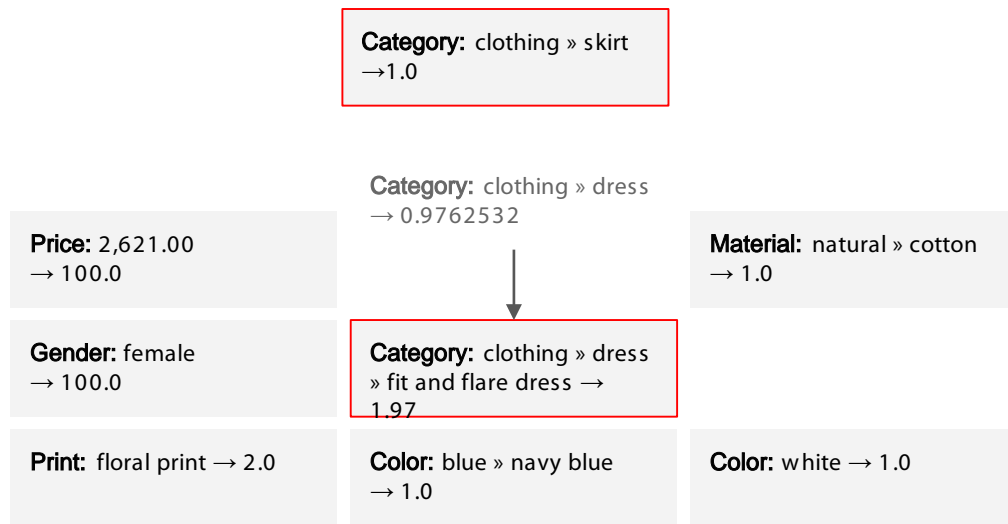
**Color:** blue » navy blue  
→ 1.0

**Material:** natural » cotton  
→ 1.0



## 2. Example: Incorrect Input

### Resolve Conflicts



## 2. Example: Incorrect Input

Assemble Final Predicted Features

**Gender:** female  
→ 100.0

**Material:** natural » cotton  
→ 1.0

**Price:** 2,621.00  
→ 100.0

**Category:** clothing » dress  
» fit and flare dress →  
1.97

**Print:** floral print → 2.0

**Color:** blue » navy blue  
→ 1.0

**Color:** white → 1.0



# 3. Data Labelling

Text and Image Annotation for Machine Learning Datasets

Tools and Services:

- Web Service: LabelBox (<https://www.labelbox.io/>)
- Web Service: Supervisely (<https://supervise.ly/>)
- Web Service: Sequence.work (<https://sequence.work/>)
- Mac Application: RectLabel (<https://rectlabel.com/>)
- **Mechanical Turk**

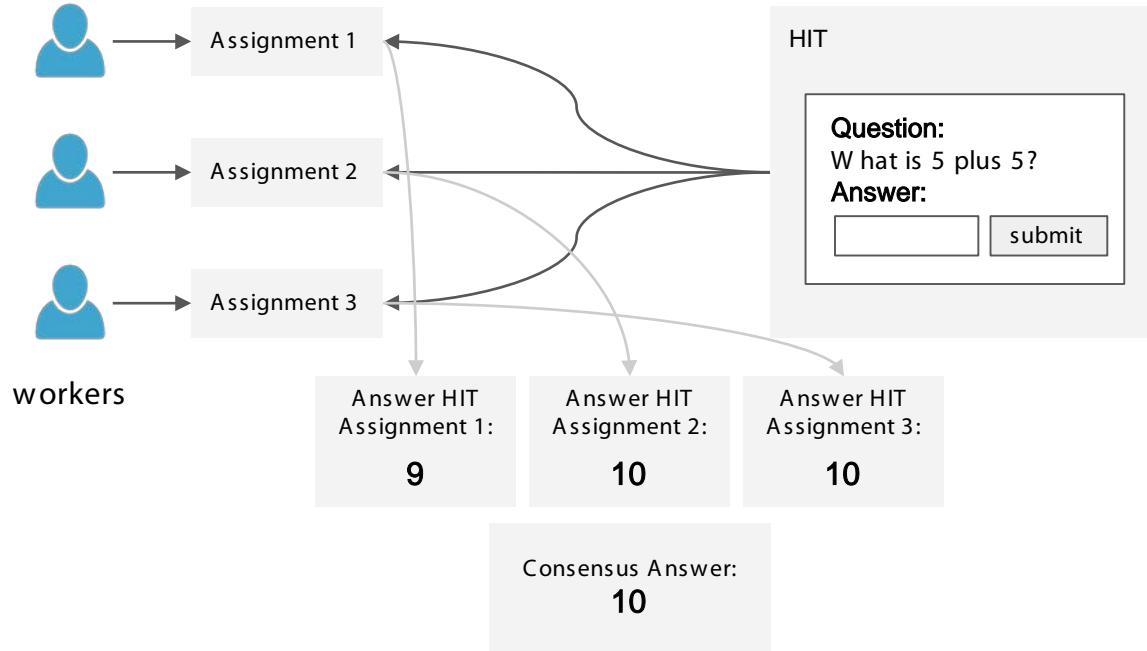
Excellent Resource:

[https://en.wikipedia.org/wiki/List\\_of\\_manual\\_image\\_annotation\\_tools](https://en.wikipedia.org/wiki/List_of_manual_image_annotation_tools)



# 3. Mechanical Turk (MTurk) - Overview

## Assignments and Human Intelligence Tasks (HITs) - Consensus

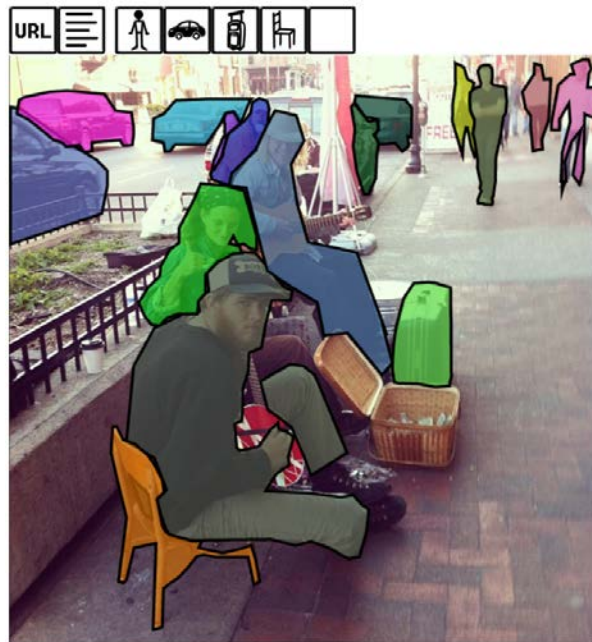


# 3. COCO Dataset

## Example: COCO (Common Objects in Context)

A large-scale object detection, segmentation and captioning dataset with:

- 123,287 images
- 886,284 instances
- 91 object types
- created using mechanical turk

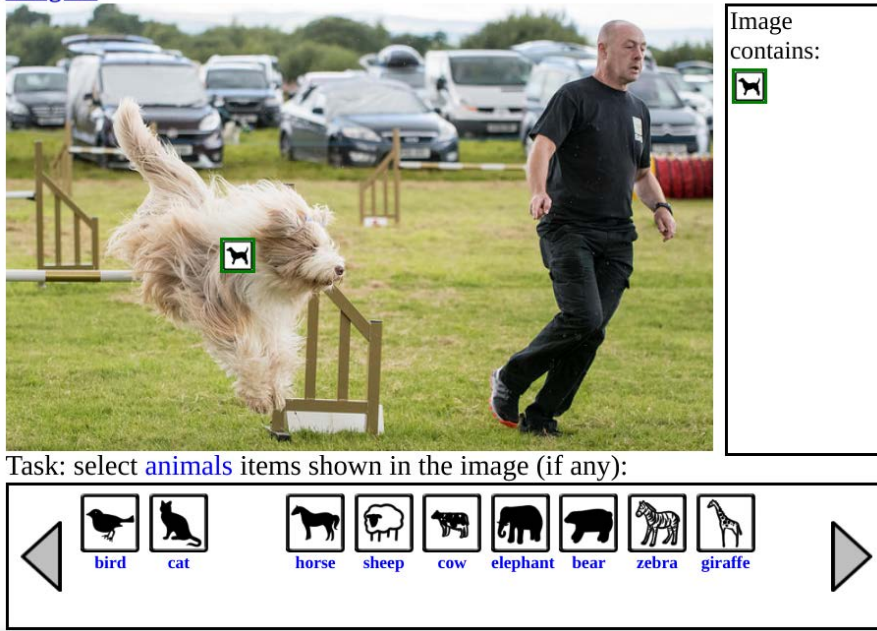




# 3. COCO Dataset

Example: COCO (Common Objects in Context)

Image 1:



↪ MS COCO Annotation UI, <https://github.com/tylin/coco-ui>



# Supported By

