Fast Object Class Labelling via Speech

Michael Gygli

gyglim@google.com

Vittorio Ferrari

vittoferrari@google.com

Google Research

Abstract

Object class labelling is the task of annotating images with labels on the presence or absence of objects from a given class vocabulary. Simply asking one yes/no question per class, however, has a cost that is linear in the vocabulary size and is thus inefficient for large vocabularies. Modern approaches rely on a hierarchical organization of the vocabulary to reduce annotation time, but remain expensive (several minutes per image for the 200 classes in ILSVRC). Instead, we propose a new interface where classes are annotated via speech. Speaking is fast and allows for direct access to the class name, without searching through a list or hierarchy. As additional advantages, annotators can simultaneously speak and scan the image for objects, the interface can be kept extremely simple, and using it requires less mouse movement. As annotators using our interface should only say words from a given class vocabulary, we propose a dedicated task that trains them to do so. Through experiments on COCO and ILSVRC, we show our method yields high-quality annotations at $2.3\times - 14.9\times$ less annotation time than existing methods.

1. Introduction

Deep neural networks need millions of training examples to obtain high performance. Therefore, large and diverse datasets such as ILSVRC [6], COCO [16] or Open Images [14] lie at the heart of the breakthrough and ongoing advances in visual recognition.

Datasets for recognition are typically annotated in two stages [6, 14, 16, 30] (Fig. 1): (i) determining the presence or absence of object classes in each image, and (ii) providing bounding boxes or segmentation masks for all classes present. Our work focuses on the former, which we call *object class labelling*. As marking a class as present requires finding at least one object of that class, we also ask annotators to click on it (as also done for the COCO dataset [16]). This task is not only natural, it also helps the subsequent annotation stages [16], and can be used as input to weakly-supervised methods [1, 18, 19, 25].

Object class labelling has traditionally been time-
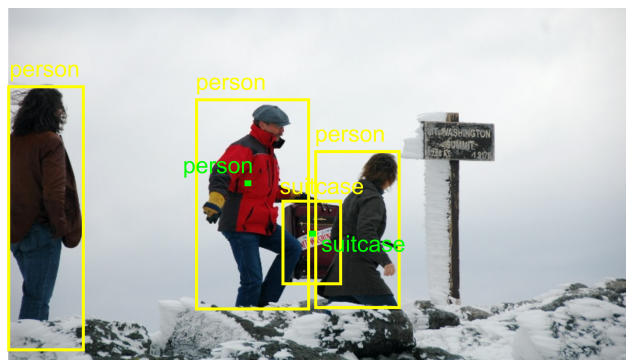
Object class labels: person, suitcase

Figure 1: Illustration of common stages of image annotation: typically annotators first provide object class labels at the image-level [6, 14] (red), sometimes associated to a specific object via a click as in [16] and our approach (green). Following stages then annotate the spatial extent of objects, *e.g.* with bounding boxes or segmentations (yellow).

consuming for annotators. A naïve approach is to ask a separate yes/no question for each class of a given vocabulary. Such a protocol is rooted on the vocabulary, not the image content. It scales linearly in the size of the vocabulary, even when only few of the classes are present in the image (which is the typical case). Thus, it becomes very inefficient when the vocabulary is large. Let's take the ILSVRC dataset as an example: getting labels for the 200 object classes in the vocabulary would take close to 6 minutes per image [13], despite each image containing only 1.6 classes on average. Previous methods have attempted to improve on this by using a hierarchical representation of the class vocabulary to quickly reject certain groups of labels [7, 16]. This reduces the annotation complexity to sub-linear in the vocabulary size. But even with these sophisticated methods, object class labelling remains time consuming. Using the hierarchical method of [7] to label the 200 classes of ILSVRC still takes 3 minutes per image [29]. The COCO dataset has fewer classes (80) and was labelled using the more efficient hierarchical method of [16]. Even so, it still took half a minute per image.

In this paper, we improve upon these approaches by us-

ing speech as an input modality. Given an image, annotators scan it for objects and mark one per class by clicking on it and saying its name. This task is rooted on the image content and naturally scales with the number of object classes in the image. Using speech has several advantages: (i) it allows for direct access to the class name via simply *saying it*, rather than requiring a hierarchical search. (ii) it does not require the experiment designer to construct a natural, intuitive hierarchy, which becomes difficult as the class vocabulary grows [28]. (iii) combining speaking with pointing is natural and efficient: when using multimodal interfaces, people naturally choose to point for providing spatial information and to speak for semantic information [22]. Also, these two tasks can be done concurrently [11, 22]. (iv) As the class label is provided via speech, the task requires less mouse movement and the interface becomes extremely simple (no need to move back and forth between the image and the class hierarchy representation). (v) Finally, speaking is fast, *e.g.* people can say 150 words per minute when describing images [31]. In comparison, people normally type at 30-100 words per minute [3, 12]. Thanks to the above points, our interface is more time efficient than hierarchical methods.

Using speech as an input modality, however, poses certain challenges. In order to reliably transcribe speech to text, several technical challenges need to be tackled, such as segmenting the speech and obtaining high-accuracy transcriptions. Furthermore, as speech is free-form in nature, annotators need to be trained to know the class vocabulary to be annotated in order to not label other objects or forget to annotate some classes. We show how to tackle these challenges and design an annotation interface that allows for fast and accurate object class labelling.

In our extensive experiments we:

- Show that speech provides a fast way for object class labelling: 2.3× faster on the COCO dataset [16] than the hierarchical approach of [16], and 14.9× faster than [7] on ILSVRC [28].
- Demonstrate the ability of our method to scale to large vocabularies.
- Show that our interface enables to carry out the task with 3× shorter mouse paths than [16].
- Show that through our training task annotators learn to use the provided vocabulary for naming objects with high fidelity.
- Analyse the accuracy of models for automatic speech recognition (ASR) and show that it supports deriving high-quality annotations from speech.

2. Related Work

Using speech as an input modality has a long history [2] and is recently emerging as a research direction in Computer Vision [4, 9, 31, 32]. To the best of our knowledge,

however, our paper is the first to show that speech allows for more efficient object class labelling than the prevailing hierarchical approaches [7, 16] . We now discuss previous works in the areas of leveraging speech, efficient image annotation and learning from point supervision.

Leveraging speech inputs. To point and speak is an efficient and natural way of human communication. Hence, this approach was quickly adopted when designing computer interfaces: as early as 1980, Bolt [2] investigates using speech and gestures for manipulating shapes. Most previous works in this space analyse what users choose when offered different input modalities [10, 21, 22, 23], while only a few approaches focus on the added efficiency of using speech. The most notable such work is [26], which measures the time needed to create a drawing in MacDraw. They compare using the tool as is, which involves selecting commands via the menu hierarchy, to using voice commands. They show that using speech gives an average speedup of 21% and mention this is a "lower bound", as the tool was not designed with speech in mind.

In Computer Vision, Vasudevan *et al.* [32] detect objects given spoken referring expressions, while Harwath *et al.* [9] learn an embedding from spoken image-caption pairs. Their approach obtains promising first results, but still performs inferior to learning on top of textual captions obtained from Google's automatic speech recognition. Damen *et al.* [5] annotates the EPIC-KITCHENS dataset based on spoken free-form narratives, which cover only some of the objects present in the image. Moreover, these narratives are transcribed *manually*, and then object class labels are derived from transcribed nouns, again manually. Instead, our approach is fully automatic and we exhaustively label all objects from a given vocabulary. Finally, more closely related to our work, Vaidyanathan *et al.* [31] re-annotated a subset of COCO with spoken scene descriptions and human gaze. While efficient, free-form scene descriptions are more noisy when used for object class labelling, as annotators might refer to objects with ambiguous names, mention nouns that do not correspond to objects shown in the image [31], or there might be inconsistencies in naming the same object classes across different annotators. Our approach avoids the additional complexities of parsing free-form sentences to extract object names and gaze data to extract object locations.

Sub-linear annotation schemes. The naïve approach to annotating the presence of object classes grows linearly with the size of the vocabulary (one binary present/absent question per class). The idea behind sub-linear schemes is to group the classes into meaningful super-classes, such that several of them can be ruled out at once. If a super-class (*e.g.* animals) is not present in the image, then one can skip the questions for all its subclasses (cat, dog, *etc.*). This grouping of classes can have multiple levels. The annotation schemes behind COCO [16] and ILSVRC [7, 28]

datasets both fall into this category, but they differ in how they define and use the hierarchy.

ILSVRC [28] was annotated using a series of hierarchical questions [7]. For each image, 17 top-level questions were asked (*e.g.* "Is there a living organism?"). For groups that are present, more specific questions are asked subsequently, such as "Is there a mammal?", "Is there a dog?", *etc*. The sequence of questions for an image is chosen dynamically, such that the they allow to eliminate the maximal number of labels at each step [7]. This approach, however, involves repeated visual search, in contrast to ours, which is guided by the annotator scanning the image for objects, done only once. Overall, this scheme takes close to 3 minutes per image [29] for annotating the 200 classes of ILSVRC. On top of that, constructing such a hierarchy is not trivial and influences the final results [28].

In the protocol used to create COCO [16], annotators are asked to mark one object for each class present in an image by choosing its symbol from a two-level hierarchy and dragging it onto the object (Fig. 4). While this allows to take the image, rather than the questions as the root of the labelling task, it requires repeatedly searching for the right class in the hierarchy, which induces significant time cost. In our interface, such an explicit class search is not needed, which speeds up the annotation process.

Rather than using a hierarchy, Open Images [14] uses an image classifier to create a shortlist of object classes likely to be present, which are then verified by annotators using binary questions. The shortlist is generated using a predefined threshold on the classifier scores. Thus, this approach trades off completeness for speed. In practice, [14] asks annotators to verify 10 out of 600 classes, but report a rather low recall of 59%, despite disregarding "difficult" objects in evaluation.

Point supervision. The output of our annotation interface is a list of all classes present in the image with a point on one object for each. This kind of labelling is efficient and provides useful supervision for several image [1,15,25] and video [18, 19] object localization tasks. In particular, [1, 18, 25] show that for their task, point clicks deliver better models than other alternatives when given the same annotation budget.

3. Speech-based annotation

We now describe our annotation task, which produces a series of time-stamped click positions $\{p_i\}$ and an audio recording for each image (Sec. 3.1). From this, we obtain object class labels by associating audio segments to clicks and then transcribing the audio (Sec. 3.2). Before annotators can proceed to the main task, we require them to pass a training stage. This helps them memorise the class vocabulary and get confident with using the interface (Sec. 3.3).

Figure 2: **Our interface.** Given an image the annotator is asked to click on one object per class and say its name. To aid memory, we additional allow to review the class vocabulary through the "Show classes" button.

3.1. Annotation task

First, annotators are presented with the class vocabulary and instructed to memorise it. Then, they are asked to label images with object classes from the vocabulary, by scanning the image and saying the names of the different classes they see. Hence, this is a simple visual search task that does not require any context switching. While we are primarily interested in object class labels, we ask annotators to click on one object for each class, as the task naturally involves finding objects anyway. Also, this brings valuable additional information, and matches the COCO protocol, allowing for direct comparisons (Sec. 4.1). Fig. 2 shows the interface with an example image.

To help annotators restrict the labels they provide to the predefined vocabulary, we allow them to review it using a button that shows all class names including their symbols.

3.2. Temporal segmentation and transcription

In order to assign class names to clicks, we need to transcribe the audio and temporally align the transcriptions. To obtain transcriptions and their start and end time we rely on Google's automatic speech recognition API[1]. While it would be possible to first transcribe the full audio recording and then match the transcriptions to clicks, we found that the temporal segmentation of transcriptions is error-prone. Hence, we opt to first segment the audio recording based on the clicks' timestamps and then transcribe these segments.

Temporal segmentation of the recording. We create an object annotation o_i for each click at position p_i and time

---

[1]https://cloud.google.com/speech-to-text/

(a)

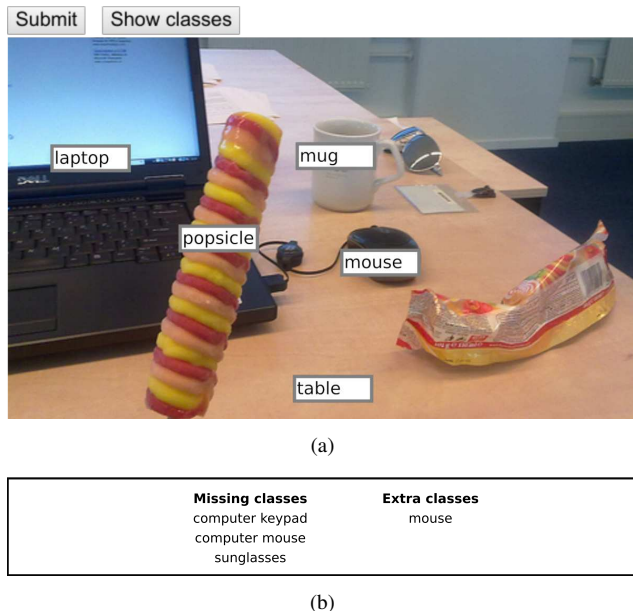| Missing classes	Extra classes
computer keypad	mouse
computer mouse	
sunglasses	

(b)

Figure 3: **Training process**. 3a shows the training task: marking an object per class with a click and saying and writing its name. 3b shows the feedback provided after each image.

t_i. For each object annotation we create an audio segment $[t_i - \delta, t_{i+1}]$, *i.e.* an interval ranging from shortly before the current click to the next click. Finally, we transcribe these audio segments and assign the result to their corresponding object annotations o_i. Empirically, using a small validation set, we found that $\delta = 0.5$s performs best, as people often start speaking slightly before clicking on the object [22].

Transcribing the object class name. The speech transcription provides a ranked list of alternatives. To find the most likely class in the vocabulary we use the following algorithm: (i) if one or more transcriptions match a class in the vocabulary, we use the highest ranking; ii) in the rare case that none matches, we represent the vocabulary and all the transcriptions using word2vec [20] and use the most similar class from the vocabulary, according to their cosine similarity. This class c_i is then treated as the label of o_i.

3.3. Annotator training

Before tackling the main task, annotators go through a training stage which provides feedback after every image and also aggregated statistics after 80 images. If they meet our accuracy targets, they can proceed to the main task. If they fail, they can repeat the training until they succeed.

Purpose of training. Training helps annotators to get confident with an interface and allows to ensure they correctly solve the task and provide high-quality labels. As a consequence, it has become common practice [14, 16, 25, 28, 30].

While we want to annotate classes from a predefined vo-

cabulary, speech is naturally free-form. In our initial experiments we found that annotators produced lower recall compared to an interface which displays an explicit list of classes due to this discrepancy. Hence, we designed our training task to ensure annotators memorise the vocabulary and use the correct object names. Indeed, after training annotators with this process they rarely use object names that are not in the vocabulary and obtain a high recall, comparable to [16] (Sec. 4.2 & 4.4).

Training procedure. The training task is similar to the main task, but we additionally require annotators to type the words they say (Fig. 3a). This allows to measure transcription accuracy and dissect different sources of error in the final class labelling (Sec. 4.4). After each image we provide immediate feedback listing their mistakes, by comparing their answers against a pre-annotated ground truth. This helps annotators memorise the class vocabulary and learn to spot all object classes (Fig. 3b). We base this feedback on the written words, rather than the transcribed audio, for technical simplicity.

Passing requirements. At the beginning of training, annotators are given targets on the minimum recall and precision they need to reach. Annotators are required to label 80 images and are given feedback after every image, listing their errors on that image, and on how well they do overall with respect to the given targets. If they meet the targets after labelling 80 images, they successfully pass training. In case of failure, they are allowed to repeat the training as many times as they want.

4. Experiments

Here we present experiments on annotating images using our speech-based interface and the hierarchical interface of [16]. First, in Sec. 4.1 we reimplement the interface of [16] and compare it to the official reported results in [16]. Then, we compare our interface to that of [16] on the COCO dataset, where the vocabulary has 80 classes (Sec. 4.2). In Sec. 4.3 we scale up annotation to a vocabulary of 200 classes by experimenting on the ILSVRC dataset. Finally, Sec. 4.4 provides additional analysis such as the transcription and click accuracy as well as response times per object.

4.1. Hierarchical interface of [16]

In the interface used for COCO [16], annotators are asked to mark one object for each class present in an image by choosing its symbol from a two-level hierarchy and dragging it onto the object. While [16] provides coarse timings, we opted to re-implement their interface for fair comparison and to do a detailed analysis on how annotation time is spent (Fig. 4). First, we made five crowd workers pass a training task equivalent to that used for our interface (Sec. 3.3). Then, they annotated a random subset of 300 im-
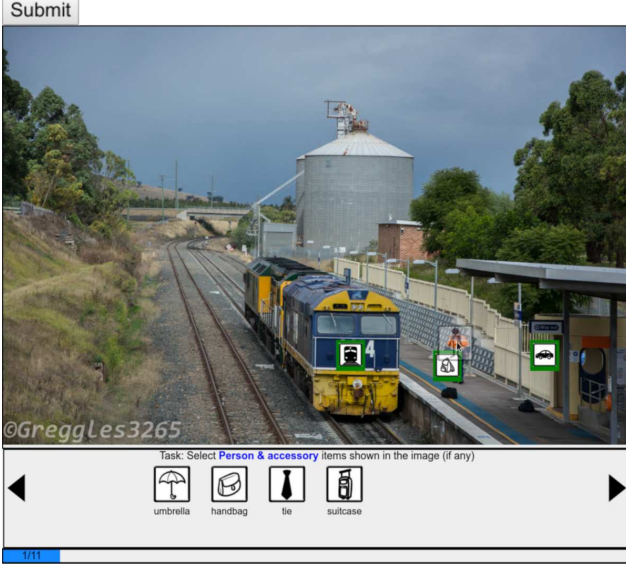
Figure 4: Our reimplementation of the hierarchical interface of [16].

Figure 5: Our approach *vs.* the hierarchical interface of [16]. Each point in the plot corresponds to an individual annotator. F1 score is the harmonic mean between recall and precision. Dataset: COCO.

ages of the COCO validation set (each image was annotated by all workers).

Results. Annotators take 29.9 seconds per image on average, well in line with the 27.4 seconds reported in [16]. Hence, we can conclude that our implementation is equivalent in terms of efficiency.

Annotators have produced annotations with 89.3% precision and 84.7% recall against the ground-truth (Tab. 1). Thus, they are accurate in the labels they provide and recover most object classes. We also note that the COCO ground-truth itself is not free of errors, hence limiting the maximal achievable performance. Indeed, our recall and precision are comparable to the numbers reported in [16].

Time allocation. In order to better understand how annotation time is spent, we recorded mouse and keyboard events. This allows us to estimate the time spent on searching for the right object class in the hierarchy of symbols and measure the time spent dragging the symbol. On average, search time is 14.8s and drag time 3.4s per image. Combined, these two amount to 61% of the total annotation time, while the rest is spent on other tasks such as visual search. This provides a target on the time that can be saved by avoiding these two operations, as done in our interface. In the remainder of this section, we compare our speech-based approach against this annotation method.

4.2. Our interface on COCO

In this section we evaluate our approach and compare it to [16]. Annotations with our interface were done by a new set of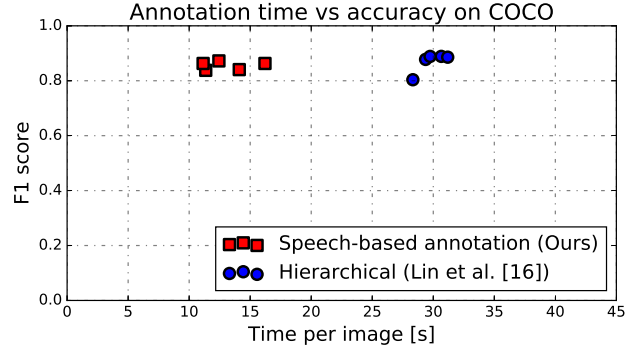 crowd workers, to avoid bias arising from having used the hierarchical interface before. The workers are all Indian nationals and speak English with an Indian accent. Hence, we use a model of Indian English for the automatic speech recognition. We also provide the class vocabulary as phrase hints[2], which is crucial for obtaining high transcription accuracy of these phrases (Sec. 4.4).

Speed and semantic accuracy. Fig. 5 and Tab. 1 show results. Our method provides a speed-up of $2.3\times$ over [16] at similar F1 scores (harmonic mean of precision and recall). In Sec. 4.1 we estimated that annotation could be sped up by up to $2.6\times$ by avoiding symbol search and dragging. Interestingly, our interface provides a speedup close to this target, confirming its high efficiency.

Despite the additional challenges of handling speech, average precision is only 2% lower than for [16]. Hence, automatic speech transcription does not affect label quality much (we study this further in Sec. 4.4). Recall is almost identical (0.8% lower), confirming that, thanks our training task (Sec. 3.3), annotators remember what classes are in the vocabulary.

Location accuracy. We further evaluate the location accuracy of the clicks by using the ground-truth segmentation masks of COCO. Specifically, given an object annotation o_i with class c_i, we evaluate whether its click position p_i lies on a ground-truth segment of class c_i. If class c_i is not present in the image at all, we ignore that click in the evaluation to avoid confounding semantic and location errors.

This analysis shows that our interface leads to high location accuracy: 96.0% of the clicks lie on the object. For the hierarchical interface it is considerably lower at 90.7%. While this may seems surprising, it can be explained by the differences in the way the location is marked. In our interface one directly *clicks* on the object, while [16] requires

---

[2]https://cloud.google.com/speech-to-text/docs/basics#phrase-hints

	Speech	Lin *et al.* [16]	Deng *et al.* [7]
COCO			
Recall	83.9 %	84.7 %	
Precision	87.3 %	89.3 %	
Time / image	13.1s	29.9s	
Time / label	4.5s	11.5s	
ILSVRC			
Recall	83.4 %	88.6 %	
Precision	80.5 %	76.6 %	
Time / image	12.0s	31.1s	≈ 179s [29]
Time / label	7.5s	18.4s	≈ 110s [29]

Table 1: Accuracy and speed of our interface (Speech) and hierarchical approaches [7, 16]. Our interface is significantly faster at comparable label quality.

Figure 6: Our approach *vs.* the hierarchical interface [16]. Each point in the plot corresponds to an individual annotator. Dataset: LSVRC.

dragging a relatively large, semi-transparent class symbol onto it (Fig. 4).

Parts of the speed gains of our interface are due to concurrently providing semantic and location information. However, this could potentially have a negative effect on click accuracy. To test this, we compare to the click accuracy that the annotators in [1] obtained on the PASCAL VOC dataset. Their clicks have a location accuracy of 96.7% comparable to our 96.0%, despite the simpler dataset with larger objects on average, compared to COCO. Hence, we can conclude that clicking while speaking does not negatively affect location accuracy.

4.3. Our interface on ILSVRC 2014

Here we apply our interface and the hierarchical interface of [16] to a larger vocabulary of 200 classes, using 300 images from the validation set of ILSVRC [28]. For [16] we manually constructed a two-level hierarchy of symbols, based on the multiple hierarchies provided by [28]. The hierarchy consists of 23 top-level classes, such as "fruit" and "furniture", each containing between 5 to 16 object classes.

Speed and semantic accuracy. Fig. 6 shows a comparison to [16] in terms of speed and accuracy, while Fig. 10 shows example annotations obtained with our interface. In Tab. 1, we also compare to the speed of [7], the method that was used to annotate this dataset. Our approach is substantially faster than both: 2.6× faster than [16] and 14.9× faster than [7]. We also note that [7] only produces a list of classes present in an image, while our interface and [16] additionally provide the location of one object per class.

Despite the increased difficulty of annotating this dataset, which has considerably more classes than COCO, our interface produces high-quality labels. The F1 score is similar to that of [16] (81.9% *vs.* 82.2%). While recall is lower for our interface, precision is higher.
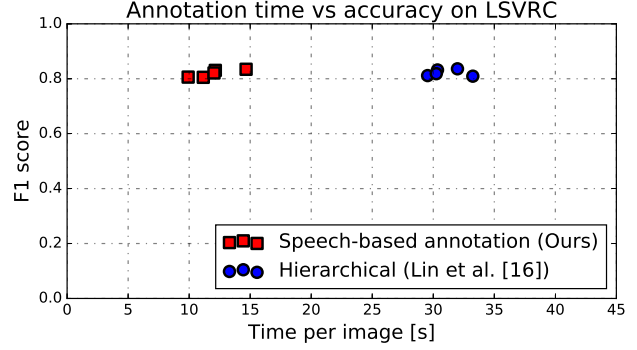
Fig. 7 shows a histogram of the annotation time per image. Most images are annotated extremely fast, despite the large vocabulary, as most images in this dataset contain few classes. Indeed, there is a strong correlation between the number of object classes present in an image and its annotation time (rank correlation 0.55). This highlights the advantage of methods that are rooted on the image content, rather than the vocabulary: their annotation time is low for images with few classes. Instead, methods rooted on the vocabulary cannot exploit this class sparsity to a full extent. The naïve approach of asking one yes-no questions per class is actually even slower the fewer objects are present, as determining the absence of a class is slower than confirming its presence [8].

4.4. Additional analysis of our interface

Time allocation. To understand how much of the annotation time is spent on what, we analyse timings for speaking and moving the mouse on the ILSVRC dataset. Of the total annotation time, 26.7% is spent on speaking. The mouse is moving 74.0% of the total annotation time, and 62.4% of the time during speaking. The rather high percentage of time the mouse moves during speaking confirms that humans can naturally carry out visual processing and speaking concurrently.

In order to help annotators label the correct classes, we allowed them to consult the class vocabulary, through a button on the interface (Fig. 2). This takes 7.2% of the total annotation time, a rather small share. Annotators consult the vocabulary in fewer than 20% of the images. When they consulted it, they spent 7.8 seconds looking at it, on average. Overall, this shows the annotators feel confident about the class vocabulary and confirms that our annotator training stage is effective.

In addition, we analyse the time it takes annotators to say an object name in Fig. 8, which shows a histogram of
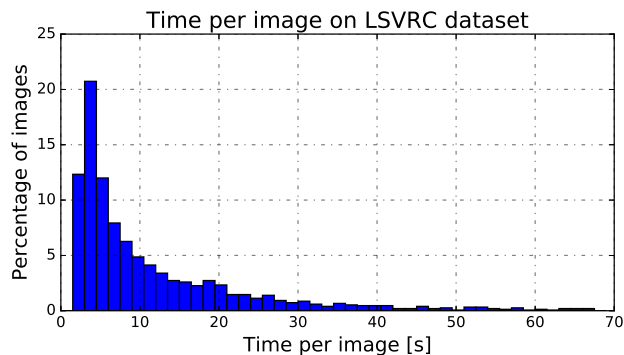
Figure 7: Histogram of the time required to annotate an image using our interface. Dataset: ILSVRC.
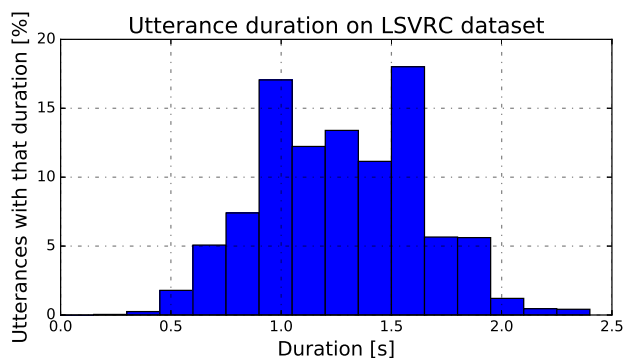

Figure 8: Histogram of the time spent *saying* the object name on ILSVRC. Saying the object names is fast and usually takes less than 2 seconds.

speech durations. As can be seen, most names are spoken in 0.5 to 2 seconds.

Per-click response time. In Fig. 9 we analyse the time taken to annotate the first and subsequent classes of an image in the COCO dataset. It takes 3.3s to make the first click on an object, while the second takes 2.0s only. This effect was also observed by [1]. Clicking on the first object incurs the cost of the initial visual search across the whole scene, while the second is a continuation of this search and thus cheaper [17, 27, 33]. After the second class, finding more classes becomes increasingly time-consuming again, as large and salient object classes are already annotated. Indeed, we find that larger objects are typically annotated first: object size has a high median rank correlation with the annotation order (-0.80). Interestingly, on the interface of [16], this effect is less pronounced (-0.50), as the annotation order is affected by the symbol search and grouping of classes in the hierarchy. Finally, our analysis shows that the annotators spent 3.9s between saying the last class name and submitting the task, indicating that they do a thorough
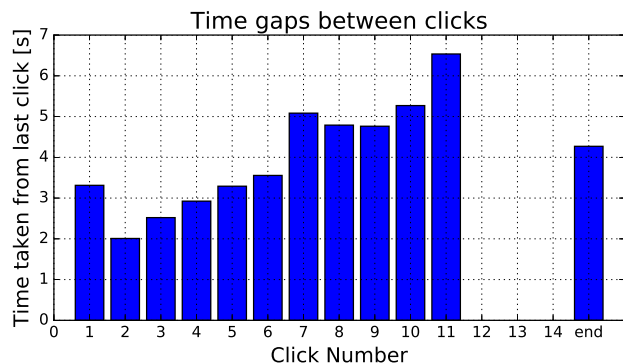

Figure 9: Analysis of the time it takes for the first and subsequent clicks when annotating object classes on the COCO dataset.

final scan of the image to ensure they do not miss any class.

Mouse path length. To better understand the amount of work required to annotate an image we also analysed the mean length of the mouse path. We find that on ILSVRC annotators using [16] move the mouse for a $3.0\times$ greater length than annotators using our interface. Thus, our interface is not only faster in terms of time, but is also more efficient in terms of mouse movements. The reason is that the hierarchical interface requires moving the mouse back and forth between the image and the class hierarchy (Fig. 11). The shorter mouse path indicates the simplicity and improved ease of use of our interface.

Training time. Training annotators to achieve good performance on the 200 classes of ILSVRC takes 1.6 hours for our interface, or 1 hour with the hierarchical interface of [16]. Instead, annotating the full ILSVRC dataset takes 1726 hours with our interface *vs.* 4474 hours with [16]. Hence, the cost of training is negligible and our interface is far more efficient than [16] even after taking training into account.

Transcription accuracy. The annotator training task provides spoken and written class names for each annotated object (Sec. 3.3). Using this data we evaluate the accuracy of the automatic speech recognition (ASR). For this we only take objects into account if they have transcriptions results attached. This keeps the analysis focused on transcription accuracy by ignoring other sources of errors, such as incorrect temporal segmentation or annotators simply forgetting to say the class name after they click on an object.

Tab. 2 shows the transcription accuracy in two setups: with and without using the vocabulary as phrase hints. Phrase hints allow to indicate phrases or words that are likely to be present in the speech and thus help the ASR model transcribe them correctly more often. Using phrase hints is necessary to obtain high transcription accuracy. Thanks to them, Recall@3 is at 96.5% on COCO and 97.5%

Figure 10: Example annotations on ILSVRC. For each click we show the three alternatives from the ASR model (orange) and the final class label (green). The first three images show typical annotations produced by our method. The last one shows a failure case: while the correct name is among the alternatives, an incorrect transcription matching a class name ranks higher, hence the final class label is wrong.
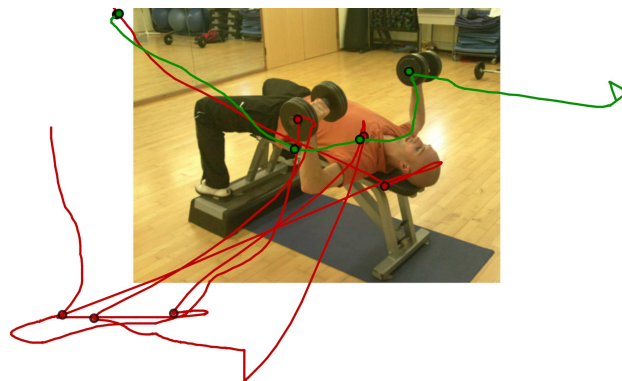

Figure 11: A comparison of typical mouse paths produced when annotating an image with our interface (green) or with [16] (red). Circles indicate clicks. Mouse paths for our interface are extremely short, thanks to its simplicity and naturalness.

	Recall@1	**Recall@3**
COCO w/ hints	93.1 %	96.5 %
COCO w/o hints	70.5 %	84.7 %
ILSVRC w/ hints	93.3 %	97.5 %
ILSVRC w/o hints	70.2 %	89.5 %

Table 2: **Transcription accuracy.** Accuracy is high when using phrase hints (see text).

on ILSVRC. Hence, the top three transcriptions usually contain the correct class name, which we then extract as described in Sec. 3.2.

In fact, we actually consider above numbers to be a lower bound on the transcription accuracy in the main task, as here we compare the transcriptions against the raw written class names, which contain a few spelling mistakes. Moreover, here the annotators are in the training phase and hence still learning about the task. Overall, the above evidence shows that ASR provides high accuracy, definitely good enough for labelling object class names.

Vocabulary usage. As speech is naturally free-form, we are interested in knowing how often annotators use object names that are outside of the vocabulary. Thus, we analyse how often the written class name in the annotator training task does not match a vocabulary name. We find that on COCO annotators are essentially only using names from the vocabulary (99.5% of the cases). On ILSVRC they still mostly use names from the vocabulary, despite the greater number of classes which induces a greater risk of misremembering their names (96.3% are in vocabulary).

Some of the out-of-vocabulary names are in fact variations of names in the vocabulary. These cases can be

mapped to their correct name in the vocabulary as described in Sec. 3.2. For example, for the ILSVRC dataset some annotators say "oven", which gets correctly mapped to "stove", and "traffic signal" to "traffic light". In other cases the annotators use out-of-vocabulary names because they actually label object classes that are not in the vocabulary (*e.g.* "fork" and "rat", which are not classes of ILSVRC).

We find that our annotator training task helps reducing the use of out-of-vocabulary names: on ILSVRC the use of vocabulary names increases from 96.3% in training to 97.5% in the main task.

5. Conclusion

We proposed a novel approach for fast object class labelling, a task that has traditionally been very time consuming. At the core of our method lies speech: annotators label images simply by saying the names of the object classes that are present. In extensive experiments on COCO and ILSVRC we have shown the benefits of our method: it offers considerable speed gains of $2.3\times - 14.9\times$ over previous methods [7, 16]. Finally, we have conducted a detailed analysis of our and previous interfaces, hence providing helpful insights for building efficient annotations tools.

We believe that speech will be useful for other tasks that combine annotating semantic and geometric properties, because speaking and moving the mouse can naturally be done concurrently [22]. In fact, our ongoing work shows that when annotating bounding boxes, class labels can be annotated without additional cost.

References

[1] A. Bearman, O. Russakovsky, V. Ferrari, and L. Fei-Fei. What's the point: Semantic segmentation with point supervision. In *ECCV*, 2016. 1, 3, 6, 7

[2] R. A. Bolt. "Put-that-there": Voice and gesture at the graphics interface. In *SIGGRAPH*, 1980. 2

[3] E. Clarkson, J. Clawson, K. Lyons, and T. Starner. An empirical study of typing rates on mini-qwerty keyboards. In *CHI*, 2005. 2

[4] D. Dai. *Towards Cost-Effective and Performance-Aware Vision Algorithms*. PhD thesis, ETH Zurich, 2016. 2

[5] D. Damen, H. Doughty, G. Maria Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling Egocentric Vision: The EPIC-KITCHENS Dataset. In *ECCV*, 2018. 2

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1

[7] J. Deng, O. Russakovsky, J. Krause, M. S. Bernstein, A. Berg, and L. Fei-Fei. Scalable multi-label annotation. In *CHI*, 2014. 1, 2, 3, 6, 8

[8] K. A. Ehinger, B. Hidalgo-Sotelo, A. Torralba, and A. Oliva. Modelling search for people in 900 scenes: A combined source model of eye guidance. *Visual cognition*, 2009. 6

[9] D. Harwath, A. Recasens, D. Surís, G. Chuang, A. Torralba, and J. Glass. Jointly Discovering Visual Objects and Spoken Words from Raw Sensory Input. In *ECCV*, 2018. 2

[10] A. G. Hauptmann. Speech and gestures for graphic image manipulation. *ACM SIGCHI*, 1989. 2

[11] D. Kahneman. *Attention and effort*. Citeseer, 1973. 2

[12] C.-M. Karat, C. Halverson, D. Horn, and J. Karat. Patterns of entry and correction in large vocabulary continuous speech recognition systems. In *ACM SIGCHI*. ACM, 1999. 2

[13] R. A. Krishna, K. Hata, S. Chen, J. Kravitz, D. A. Shamma, L. Fei-Fei, and M. S. Bernstein. Embracing error to enable rapid crowdsourcing. In *CHI*, 2016. 1

[14] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, T. Duerig, and V. Ferrari. The Open Images Dataset V4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 1, 3, 4

[15] I. H. Laradji, N. Rostamzadeh, P. O. Pinheiro, D. Vazquez, and M. Schmidt. Where are the blobs: Counting by localization with point supervision. *arXiv preprint arXiv:1807.09856*, 2018. 3

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 1, 2, 3, 4, 5, 6, 7, 8

[17] A. Lleras, R. A. Rensink, and J. T. Enns. Rapid resumption of interrupted visual search: New insights on the interaction between vision and memory. *Psychological Science*, 2005. 7

[18] S. Manen, M. Gygli, D. Dai, and L. Van Gool. PathTrack: Fast Trajectory Annotation with Path Supervision. In *ICCV*, 2017. 1, 3

[19] P. Mettes, J. C. van Gemert, and C. G. Snoek. Spot on: Action localization from pointly-supervised proposals. In *ECCV*, 2016. 1, 3

[20] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 4

[21] S. Oviatt. Multimodal interfaces for dynamic interactive maps. In *ACM SIGCHI*, 1996. 2

[22] S. Oviatt. Multimodal interfaces. *The human-computer interaction handbook: Fundamentals, evolving technologies and emerging applications*, 2003. 2, 4, 8

[23] S. Oviatt, A. DeAngeli, and K. Kuhn. Integration and synchronization of input modes during multimodal human-computer interaction. In *CHI*, 1997. 2

[24] D. P. Papadopoulos, A. D. F. Clarke, F. Keller, and V. Ferrari. Training object class detectors from eye tracking data. In *ECCV*, 2014.

[25] D. P. Papadopoulos, J. R. Uijlings, F. Keller, and V. Ferrari. Training object class detectors with click supervision. In *CVPR*, 2017. 1, 3, 4

[26] R. Pausch and J. H. Leatherby. An empirical study: Adding voice input to a graphical editor. *J. American Voice Input/Output Society*, 1991. 2

[27] K. Rayner. Eye movements and attention in reading, scene perception, and visual search. *Quarterly Journal of Experimental Psychology*, 2009. 7

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 2015. 2, 3, 4, 6

[29] O. Russakovsky, L.-J. Li, and L. Fei-Fei. Best of both worlds: human-machine collaboration for object annotation. In *CVPR*, 2015. 1, 3, 6

[30] H. Su, J. Deng, and L. Fei-Fei. Crowdsourcing annotations for visual object detection. In *AAAI Human Computation Workshop*, 2012. 1, 4

[31] P. Vaidyanathan, E. Prud, J. B. Pelz, and C. O. Alm. SNAG : Spoken Narratives and Gaze Dataset. *ACL*, 2018. 2

[32] A. B. Vasudevan, D. Dai, and L. Van Gool. Object Referring in Visual Scene with Spoken Language. In *CVPR*, 2017. 2

[33] D. G. Watson and M. Inglis. Eye movements and time-based selection: Where do the eyes go in preview search? *Psychonomic Bulletin & Review*, 2007. 7