SPAMming Labels: Efficient Annotations for the Trackers of Tomorrow

Orcun Cetintas [1*]   **Tim Meinhardt** [2]   **Guillem Brasó** [1*]   **Laura Leal-Taixé** [2]
[1] Technical University of Munich   [2] NVIDIA

Abstract

Increasing the annotation efficiency of trajectory annotations from videos has the potential to enable the next generation of data-hungry tracking algorithms to thrive on large-scale datasets. Despite the importance of this task, there are currently very few works exploring how to efficiently label tracking datasets comprehensively. In this work, we introduce SPAM, a tracking data engine that provides high-quality labels with minimal human intervention. SPAM is built around two key insights: i) most tracking scenarios can be easily resolved. To take advantage of this, we utilize a pre-trained model to generate high-quality pseudo-labels, reserving human involvement for a smaller subset of more difficult instances; ii) handling the spatiotemporal dependencies of track annotations across time can be elegantly and efficiently formulated through graphs. Therefore, we use a unified graph formulation to address the annotation of both detections and identity association for tracks across time. Based on these insights, SPAM produces high-quality annotations with a fraction of ground truth labeling cost. We demonstrate that trackers trained on SPAM labels achieve comparable performance to those trained on human annotations while requiring only $3 - 20\%$ of the human labeling effort. Hence, SPAM paves the way towards highly efficient labeling of large-scale tracking datasets. Our code and models will be available upon acceptance.

1 Introduction

Detecting and tracking objects in time is at the core of many challenging vision applications. To address these problems, modern multi-object tracking (MOT) approaches [46, 79, 81, 75, 69] rely on increasingly larger amounts of annotated data. While annotating image datasets is already costly, introducing a temporal component further increases the task difficulty and data scale requirements. This is due to the fact that redundancies between frames cause the information density to scale poorly with the amount of data, making the overall annotation task more challenging and resource-intensive.

To mitigate these issues, researchers are spending more effort in operating with limited human annotations. Self- and semi-supervised learning (SSL) and pseudo-labeling (PL) approaches are often used to leverage unlabeled data whenever possible [36, 42, 35], while Active learning (AL) strategies aim at selecting the most informative samples to label. All these strategies are commonly applied in various image understanding tasks, such as classification [14, 11] or object detection [17, 70]. Even if video tasks do not have any lower requirement for data, there are only a handful of works that target efficient labeling in the video domain [62, 63, 64, 40]. Existing works fail to provide *comprehensive* solutions to video labeling by either ignoring its dense temporal component [40, 64] or working under a limited single-object setup [74]. An effective MOT labeling engine should operate across multiple objects and frames, considering the intricate dependencies between tracks inherent in the labeling problem.

---

* Work done during an internship at NVIDIA.

Videos have unique characteristics that can be leveraged to unleash the full potential of a modern MOT label engine. Firstly, frame-wise similarities in videos yield data redundancies. In the context of MOT, this implies that some associations between detections are relatively easy to solve [7]. This setup provides an ideal opportunity to leverage state-of-the-art models pre-trained to pseudo-label new data at zero cost. Secondly, the dependencies of objects across time and space cause annotations in one frame of the video to have cascading effects in other frames. For example, solving an association for one track can resolve associations for neighboring tracks. In order to maximize the efficiency of the annotations, we frame the labeling task within a model that takes into account such dependencies. Thus, we adopt a track-centric labeling approach rather than a frame-centric one [40].

We present the **SPAM** video data label engine, which chains **S**ynthetic pre-training, **P**seudo-labeling, and **A**ctive learning with a graph-based **M**odel for optimal tracking performance with minimal human effort. In the image domain, there are plenty of large-scale annotated datasets that a model-driven approach can leverage for pre-training [35], but this is not the case for multi-object tracking datasets, which tend to be much smaller in scale. To avoid depending on an initial well-curated large-scale dataset, we advocate for a **S**ynthetic pre-training. We further analyze which components of a tracking method are more affected by the synthetic-to-real gap. This first experiment gives us an important insight: labeling should be focused on detections and associations, and we can directly use the synthetically trained reID model. With our synthetic pre-trained model, we then propose a model-driven approach to produce **P**seudo-labels on real data and retrain the model with its own pseudo-labels. We show that with this recipe we can even reach a performance comparable to ground truth without ever training our labeler on manually annotated data.

Synthetic pre-training and pseudo-labeling allow us to handle easy decisions and defer only uncertain or complex decisions to an annotator. We propose an **A**ctive learning strategy where the labeler makes decisions at the track-level instead of frame-level, thereby utilizing the annotation budget more efficiently. Towards this goal, we solve the labeling problem with a **M**odel, based on graph hierarchies [12]. This model captures long-range spatio-temporal dependencies, which means our annotations will have an impact across frames, and not only on the frame where the annotation actually happens. We further enhance the model with a detection filtering layer so that we can naturally label detection and association within the same framework.

In summary, we propose **SPAM**, a multi-object tracking data label engine that significantly reduces annotation cost by dealing with easy scenarios with (i) a strong MOT model pre-trained on **S**ynthetic data, and used to (ii) **P**seudo-label real data for its own re-training. Harder detection and association decisions are carefully selected by an (iii) **A**ctive learning scheme that works on a (iv) **M**odel based on graph hierarchies. This allows us to efficiently use the annotator's budget and account for spatio-temporal dependencies in the labels. The resulting model generates autolabels comparable to ground truth performance, requiring only $3 - 20\%$ of manual annotation effort across three diverse datasets.

2   Related Work

As a comprehensive video data label engine, SPAM is the first of its kind to label tracking datasets by uniquely chaining the following four aspects.

Pre-training for tracking. Over the years, the multi-object tracking (MOT) community has annotated many challenging benchmark datasets [18, 19, 60]. However, due to the significant costs associated with manually labeling temporal data, most benchmarks only provide few annotated video sequences. To obtain top performance, modern data-hungry tracking methods are usually pre-trained on additional video and tracking datasets [22, 77, 20, 71, 80]. Many works also rely on relevant image recognition [59] datasets to boost their per-frame object detection performance. An alternative pre-training direction avoids human annotations entirely by relying on simulated synthetic data [26, 24, 61, 29, 23].

Our label engine also pre-trains the labeling model synthetically, minimizing its total annotation footprint. To this end, we demonstrate effectiveness of synthetic pre-training [23] for generating pseudo-labels for self-supervision.

Self-supervision in the temporal domain. To benefit not only from already annotated datasets, many works explore the utilization of available unlabeled data sources, *i.e.*, how to perform self-supervised learning for videos. Notable examples leverage cycle-consistency-based correspondences [67, 30],
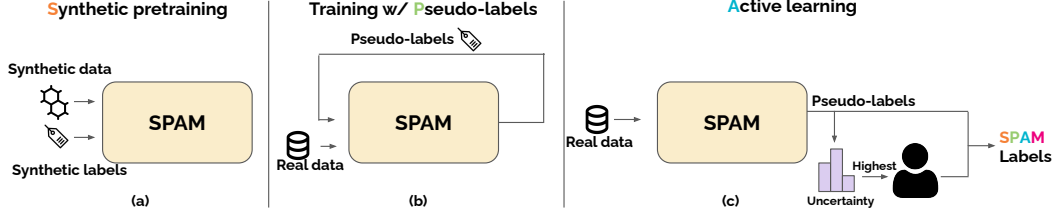
Figure 1: Overview of the SPAM training and annotation pipeline. (a) Initial model training on synthetic data. (b) Application of SPAM to generate pseudo-labels without incurring manual annotation costs on a real dataset, followed by training on pseudo-labels. (c) Real dataset labeling using pseudo-labels and an uncertainty-based active learning approach.

color propagation [66] or augmentation-based consistencies [4, 43]. Furthermore, [47] uses contrastive learning over re-identification (reID) embeddings to perform unsupervised association learning. More recently, several works [36, 42, 35, 33] in the image domain demonstrate the effectiveness of training on pseudo-labels generated by strong underlying labeling models. In the video domain, this type of self-supervision is mostly unexplored.

We demonstrate the effectiveness of pseudo-labels for the temporal domain by improving the label quality of our SPAM labeler via self-supervision on its own output. The necessary signal-to-noise ratio for this training loop is obtained by pre-training a strong graph-based model on synthetic data.

Efficient labeling of video datasets. While significant effort has been devoted towards increasing the labeling efficiency of image datasets [1, 35], similar approaches for the video and tracking domain are still relatively limited. Earlier works reduce the number of frames to annotate within individual videos but still rely heavily on human annotators [65]. Other solutions are limited to only a single object per sequence [63, 16] or still rely on human bounding box annotations [62] to label segmentation masks across time. The authors of [41] select key frames based on their expected impact on tracking performance. While effective within its scope, this approach only yields sparse frame annotations, which are insufficient for training most modern trackers. To reduce the human annotation effort for trajectories, [45] proposes an efficient video interface that still requires labeling all trajectories.

In contrast, our label engine performs active learning for multiple objects across all video frames and is the first of its kind to tackle long-term tracking problems via a hierarchical labeling approach. The uncertainty measures provided by our graph-based model allow us to focus expensive annotation efforts on individual tracks instead of labeling entire frames in a brute-force manner.

Multi-object tracking models. Tracking objects across time in a video is an established problem with a long research history. The most dominant tracking-by-detection (TbD) paradigm splits the task into two steps: (i) detecting objects in every frame and (ii) associating them to object trajectories, also referred to as data association. Modern trackers have explored performing (i) and (ii) jointly [25, 7, 46, 81], or providing solutions for (ii) relying on motion models [52, 39, 2, 55, 3, 56, 50, 72, 78], appearance models [57, 79, 49, 73, 37, 54, 13, 15] or more complex techniques involving transformers [75, 10]. An alternative direction of work relies on graph formulations [6, 31, 51, 76, 38] to utilize the TbD paradigm. In this formulation, nodes model object detections, and edges, association hypotheses. This long-studied formulation has recently been revived with remarkable success with the help of learnable graph neural network (GNN)architectures [9, 8, 68, 12].

For our label engine to generate long-term pseudo-labels and active learning uncertainty estimations, we rely on a strong graph-based tracker [12]. To incorporate both detection and association labeling within a unified graph framework, we introduce a novel hierarchy level dedicated to detection filtering.

3 SPAM

3.1 Overview

Our objective is to minimize the manual annotation effort required for multi-object tracking datasets. Our key observation is that most association cases can be solved easily by models pre-trained on synthetic data. Therefore, no human effort is needed, and we can instead use our model's predictions
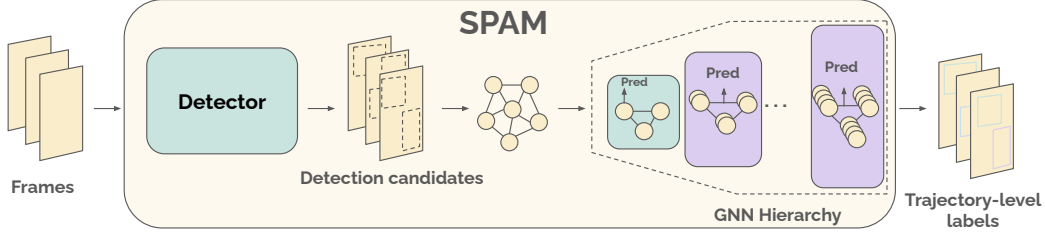
Figure 2: Overview of the SPAM model. We first generate a set of detection candidates with our detector. Hierarchical GNNs then classify these candidates into valid and invalid objects via node classification, and assign identities through edge classification.

as *pseudo-labels*. Decisions that are more complex or uncertain are be selected by an active learning scheme to be annotated manually. The resulting set of labels achieves near-ground truth (GT) performance with a dramatically reduced need for human annotation, thereby vastly enhancing the efficiency of the overall labeling process.

In order to perform pseudo-labeling and active learning, we require a model that can capture long-term spatio-temporal dependencies between tracks. We leverage a hierarchical graph neural network formulation and cast the annotation task over a graph. Our architecture draws inspiration from SUSHI [12], which demonstrated remarkable association performance. Our model's scope, however, goes beyond association. Given the constraints of the labeling task, we need to equip our model with the ability to reason about detections before performing association. To address this broader goal, our GNN hierarchy, unlike SUSHI [12], also includes the capability to accept or reject detections by classifying nodes into correct and incorrect object hypotheses.

Our overall model is pre-trained on synthetic data [23] and further fine-tuned on its own predicted pseudo-labels over the target dataset that we aim to annotate. This recipe allows for strong labelling performance *without any human annotations*, and provides the basis for a scalable video annotation engine. After this training loop, we use our updated model's predictions as pseudo-labels to annotate a large portion of the target dataset. Decisions that are more complex or uncertain are selected by an active learning scheme to be annotated manually. The resulting set of labels achieves near-GT performance with a dramatically reduced set of manual annotations, vastly enhancing the efficiency of the overall process. We provide an overview of our training and annotating recipe in Fig. 1.

3.2 Graph Hierarchies as a Model for Labeling

Problem formulation. Our model builds upon the standard graph formulation [76] where, given a set of object candidates $\mathcal{O}$, our goal is to obtain a set of valid objects $\mathcal{O}_v \subset \mathcal{O}$ and their corresponding trajectories $\mathcal{T}$. Each trajectory $T_k \in \mathcal{T}$ consists of a set of objects sharing the same identity or, equivalently, the set of edges connecting those nodes. Specifically, we model each object candidate $o_i \in \mathcal{O}$ as a node and each association hypothesis among candidates as an edge in an undirected graph $G = (V, E)$. Within this formulation, nodes $u \in V$ could be classified into valid objects if $u \in \mathcal{O}_v$. Analogously, edges can be classified into correct or incorrect hypotheses depending on whether they belong to some trajectory or not. This formulation can be generalized to a hierarchical setting by splitting video sequences into non-overlapping subsequences, *i.e.*, disjoint subgraphs, and progressively merging them. For further details we refer to the supplementary material.

Background on graph hierarchies for MOT. Our model consists of a hierarchy of GNNs operating over the graph formulation described above. Intuitively, starting from detection candidates, at each hierarchy level, GNN hierarchy subsequently merge tracklets from the previous level into longer ones. Nodes and edges are represented by their embeddings. Message passing is then used to share information within the graph in order to update node and edge embeddings with richer information. The GNNs are trained to classify edges into active and inactive association hypotheses and they share the same features, network architecture and learnable weights across hierarchy levels. In addition to the edge features reported in [12], for each object candidate o_i we concatenate its timestamp t_i, normalized bounding box coordinates and dimensions (x_i, y_i, h_i, w_i) and corresponding confidence from the detector c_i and embed it as node features.

4
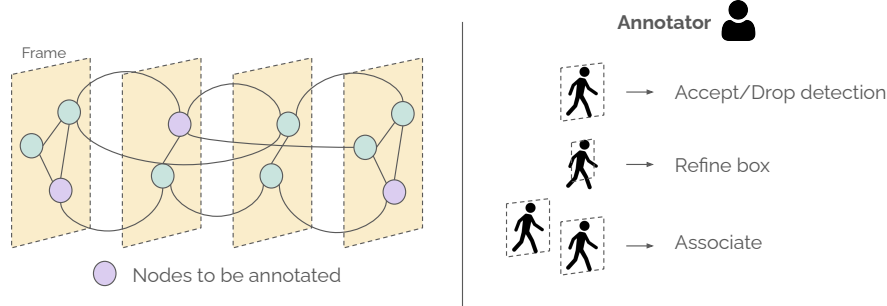
Figure 3: Our graph-based labeling pipeline begins with the selection of nodes for annotation. For each node to be labeled, the annotator is presented with the options of validating the *detection*, improving *localization* by refining the box and *association*.

Classifying nodes. We aim to minimize the annotation effort; therefore, we restrict the annotator to operate on a fixed graph of detection candidates *i.e.* no new detections are added to the graph after the initial set. This is why we have to create an initial graph that is over-complete. In order to do so, we use a low confidence threshold for our detector and obtain a large number of detection candidates (Fig. 2). While this ensures a high detection recall, it also introduces a larger number of false positive detections that our model needs to address. As we show in experiments, the original SUSHI architecture [12] cannot handle such false positives, which is why we propose a new layer in the hierarchy, GNN_{node}. Intuitively, GNN_{node} leverages temporal information to validate object candidates. We use its output node embeddings and feed them to a lightweight classifier to enable false positive detection filtering. By doing so, we can consider a large number of input proposals achieving high detection recall by leveraging the spatio-temporal reasoning ability of our GNN for detection candidates.

Pre-training and *annotation-free* fine-tuning. We pre-train all of our model components on the large-scale synthetic dataset MOTSynth [23]. To boost our model's performance and bridge the domain gap with our target dataset for annotation, we conduct further fine-tuning on the target dataset after pre-training. We achieve this by utilizing the raw predictions of our model on the target dataset and employing them as pseudo-labels for *self-training*. As we show in our experimental section, this strategy yields a significant performance improvement without incurring any human annotation cost.

3.3 How to Label on Graph Hierarchies?

Manual labeling. Labeling video data at the box level involves several steps: identifying objects of interest, localizing their boundaries with bounding boxes, and assigning them unique identities. Typically, annotators achieve this by examining each frame, detecting and localizing objects individually, and then assigning identities based on temporal context from neighboring frames. In terms of annotation effort in clicks, *detecting* and *localizing* objects typically require two clicks (one for the top-left corner and one for the bottom-right corner of the bounding box), while *associating* objects across time requires one click (identifying the next occurrence of the object in subsequent frames), as reported in [18, 19]. This manual process is labor-intensive, costly, and severely limits the availability of annotated video data.

Graph-based labeling. Utilizing a graph formulation for detection and tracking enables explicit modeling of objects in the scene and their interactions. In this framework, each node of the graph represents an object, allowing a manual annotation of an object to be framed as selecting a node on this graph. Given a node, we present the annotator with options of: accepting a *detection* and refining the bounding-box to improve *localization* (two clicks, one for the top-left corner and one for the bottom-right corner of the bounding box) and *associating* a node across time (1 click). As we are working on an over-complete set of detections, the annotator can also reject a detection (1 click). We visualize our annotation pipeline in Fig. 3.

Annotation acquisition. The core idea of our approach is to use our model decisions to guide the annotation process. For the vast majority of scenarios, we directly use our model's predictions as pseudo-labels. For a small subset of more challenging decisions where our model exhibits uncertainty,
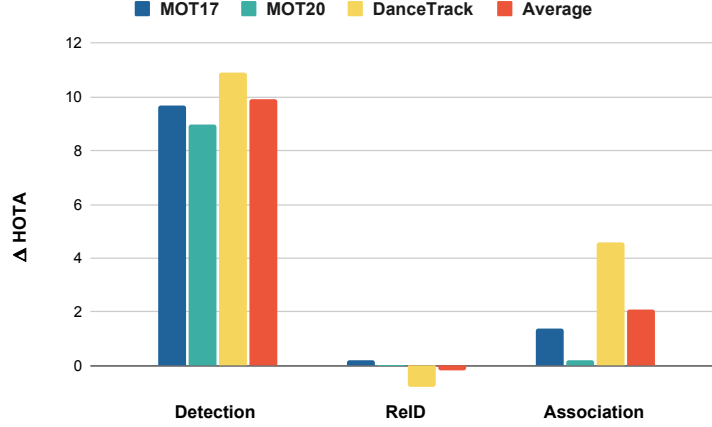
Figure 4: Analysis of performance gap between training a model on synthetic and real data for the three most common tracking components: detection, association, re-identification.

we validate our decisions with an annotator. Within our graphs, we adopt an uncertainty-based approach to identify which nodes require annotations. Specifically, for each node $v \in V$, we compute its uncertainty as $\text{uncert}(v) := \max_{u \in \mathrm{N}_v} H(\hat{y}_{(v,u)})$, where N_v denotes the set of neighboring nodes of v and $\hat{y}_{(v,u)}$ is the edge prediction for edge (v, u):

$$H(\hat{y}_{(v,u)}) := -(\hat{y}_{(v,u)} \log \hat{y}_{(v,u)} + (1 - \hat{y}_{(v,u)}) \log(1 - \hat{y}_{(v,u)})) \tag{1}$$

At the GNN_{node} level, where we have node predictions instead of edge predictions, we simply compute $\text{uncert}(v)$ as $H(\hat{y}_{(v)})$ instead. Subsequently, we refer the nodes with the highest uncertainty scores to the annotator for manual annotation and use the remaining predictions as pseudolabels for the rest of the nodes.

Hierarchical label generation. To fully leverage our hierarchical graph formulation, we apply our label acquisition approach described above in a hierarchical manner. Intuitively, we distribute our annotation budget B across L hierarchical levels as $B_1, \ldots, B_L$, with $B_1 + \cdots + B_L = B$. In deeper hierarchy levels, nodes represent tracklets instead of single detections. This allows us to propagate annotator decisions for the entire cluster instead of individual detections, resulting in solving multiple uncertainties with a single annotation. As a result, we utilize the annotation budget more effectively while also mitigating error propagation within our pseudo-labels.

4 Experiments

4.1 Datasets and Metrics

Datasets. We conduct experiments on three public benchmarks:

- **MOT17** [18] is an established benchmark containing 14 challenging video sequences, exhibiting diverse characteristics, including variations in camera motion, viewpoint, and pedestrian densities.
- **MOT20** [19] contains a total of 8 video sequences, all featuring extremely crowded scenes. Due to their high pedestrian density, the emphasis is on addressing highly challenging scenarios characterized by frequent and prolonged occlusions.
- **DanceTrack** [60] contains 100 video sequences. These sequences feature dancing individuals with similar appearances, facilitating the evaluation of tracker robustness against highly challenging motion patterns where appearance information has a relatively minor influence. This non-standard setup often leads to a performance drop in trackers [60].

Model	High Conf. Box	Low Conf. Box	GNN_{node}	MOTA ↑	IDF1 ↑	HOTA ↑	DetA ↑
A	✓	✗	✗	64.4	74.7	59.9	55.9
B	✓	✓	✗	60.6	71.4	58.5	54.9
C	✓	✓	✓	**65.4**	**75.1**	**60.4**	**56.6**

Table 1: Ablation study investigating the impact of incorporating low confidence bounding boxes with and without the GNN_{node}.

Metrics. We assess the performance of trackers using three established metrics:

- **MOTA** [32] assesses trajectory coverage, by combining object detection measurements with identity switches, with the latter playing a relatively minor role.
- **IDF1** [53] focusses on identity preservation over the sequences and provides a measure of association performance.
- **HOTA** [44] is a recently introduced metric designed to strike a balance between association and object detection performance within a single score.

4.2 Implementation Details

Architecture and training. Our model consists of three main components: (i) a detector network, based on YOLOX [27], (ii) a ResNet-based ReID network [28], and (iii) a GNN hierarchy, all pretrained on the synthetic sequences of MOTSynth [23]. Please see the supplementary material for further details about training hyperparameters. After pretraining, we use the resulting model to obtain pseudo-labels for our target MOT dataset, and then fine-tune our detector and GNN on them. Note that our Model is *never trained on human annotations*, it is only trained on synthetic annotations and its own pseudo-labels.

Experimental setup. To assess the quality of SPAM's labels, we use them to train two state-of-the-art online trackers: ByteTrack [78] and GHOST [57]. For validation, we follow the standard half-split setting [79, 78, 57] over MOT17 sequences. Annotation budgets are computed based on number of clicks as described in Section 3.3.

4.3 Synthetic Pretaining

Several works have shown that synthetic pre-training is a strong starting point for multi-object tracking methods [78, 23]. In this section, we aim to analyze the gap between training a model on synthetic data and training it with real data separately for the three most common tracking components: detection, association, re-identification. For each setting, only the specific component that we are analysing is trained on synthetic data while other components are all trained on real data. We perform this experiment on three datasets and plot the difference in HOTA when training on real data vs. synthetic data in Fig. 4 (a positive score indicates superior performance with real data training). Our first observation, is that the detector is the component that suffers the domain gap the most, with a performance drop of 9.9 HOTA points. Re-identification, on the other hand, has a similar performance when trained on synthetic or real data. Lastly, association performance is higher with real data training (2.1 HOTA). This first experiment gives us an important insight: labeling and retraining should be focused on detections and associations, and we can directly use the synthetically trained reID model.

4.4 Towards a Model for Labeling

Graph-based validation of detection candidates. In our framework, annotators have the capability to eliminate detections, but we refrain from requesting them to annotate new detections in order to maintain minimal annotation effort. Instead we start from an initial graph formed by an over-complete set of detections. Towards this end, we use a low confidence threshold for our detector, thereby reducing the number of false negatives in our initial graph and ensuring a high recall. As a result, this leads to an increase in the number of false positives incorporated into the graph, a tendency that is further accentuated by the fact that our detector is exclusively trained on synthetic data. We

Method	HOTA ↑	MOTA ↑	IDF1 ↑	HOTA ↑	MOTA ↑	IDF1 ↑	HOTA ↑	MOTA ↑	IDF1 ↑
		MOT17			MOT20			DanceTrack	
MeMOT [10]	56.9	72.5	69.0	54.1	66.1	63.7			
TrackFormer [46]	57.3	74.1	68.0	54.7	65.7	68.6			
MOTR [75]	57.8	68.6	73.4				54.2	51.5	79.7
FairMOT [79]	59.3	73.7	72.3	54.6	61.8	67.3	39.7	82.2	40.8
QDTrack [49]	63.5	78.7	77.5	60.0	74.7	73.8	54.2	87.7	50.4
ByteTrack† [78]	62.8	78.9	77.1	60.4	74.2	74.5	47.7	**89.6**	53.9
GHOST [57]	62.8	78.7	77.1	61.2	73.7	75.2	56.7	91.3	57.7
UTM [73]	64.0	**81.8**	78.7	62.5	**78.2**	76.9			
MotionTrack [52]	65.1	81.1	80.1	62.8	78.0	76.5			
SUSHI [12]	66.5	81.1	83.1	64.3	74.3	79.8	63.3	88.7	**63.4**
SPAM (Ours)	**67.5**	80.7	**84.6**	**65.8**	76.5	**81.9**	**64.0**	89.2	**63.4**

Table 2: Test set results on MOT17, MOT20 and DanceTrack when using SPAM solely as a tracker.

propose to leverage the graph structure itself to validate the detections based on temporal consistency. Specifically, we add an additional GNN-layer in our hierarchy, GNN_{node}, which intuitively examines the spatial and temporal consistency of the bounding box candidates and filters out implausible ones. In Table 1, we assess the effect of this design choice. Model A uses only high confidence bounding boxes, model B creates a graph with also low confidence bounding boxes but without GNN_{node}, while C includes GNN_{node}. As we can see, simply adding low confidence boxes results in a drastic drop in performance on all metrics. Only by adding our GNN_{node} we are able to take full advantage of the low confidence boxes without penalizing our performance with too many errors.

SPAM is a SOTA tracker. Before moving to labeling with our Model, we want to verify its overall performance and generalization capabilities when trained on real data. Only by using a state-of-the-art model we will be able to reduce annotation effort as much as possible. We therefore pre-train our model on the synthetic dataset, and finetune it on the specific real dataset using groundtruth. For a fair comparison, we obtain object detections from a YOLOX detector [27] trained following [78, 57, 12]. Our model outperforms state-of-the-art by 1.0 HOTA point on MOT17, 1.1 HOTA points on MOT20, and 0.7 HOTA points on DanceTrack as shown in Table 2. Furthermore, on MOT17 and MOT20, there is also a significant improvement in IDF1, showing the identity-preserving capabilities of the model, which is especially important for our video annotation setting. Even if being state-of-the-art is not the main goal of this paper, having a strong performing model allows us to have an excellent starting point for producing pseudolabels. We would like to emphasize that, we use ground truth annotations only for this experiment just to validate the performance of our model with a fair comparison with other trackers. From this point onward, we will not use ground truth labels. We will use our synthetic pretraining weights and focus on the pseudolabeling and active learning aspects of our method to show how to annotate multi-object tracking datasets.

4.5 Enhancing Annotation Quality through Pseudolabels and Active Learning

Retraining our Model on Pseudolabels. In this experiment, we compare two models. The first one, SPAM, is only trained on synthetic data. The second one, SPAM , is trained in three stages: (i) pre-training on synthetic data, (ii) pseudo-labeling the real dataset with our model, and (iii) re-training our model on our own pseudo-labels. In Section 4.5, we observe a significant improvement when using pseudo-labels to re-train our model: an increase of 4-6 HOTA, 4-6 MOTA and 4-7 IDF1 on MOT17, MOT20 and DanceTrack datasets. Note that these improvements come at zero human annotation effort as this process requires no manual annotations. Further pseudo-labeling the real dataset and re-training our model does not lead to another performance increase. It is therefore time to turn into manual annotations, and in particular, finding the best use of our annotator's effort.

Model	Pseudolabel	HOTA ↑	MOTA ↑	IDF1 ↑
		MOT17		
SPA**M**	✗	60.0	65.3	75.1
SPA**M**	✓	**63.8**	**69.2**	**79.7**
		MOT20		
SPA**M**	✗	52.2	65.4	70.3
SPA**M**	✓	**58.7**	**70.2**	**77.1**
		DanceTrack		
SPA**M**	✗	41.8	65.2	43.4
SPA**M**	✓	**48.1**	**71.0**	**50.2**

Table 3: Performance boost obtained by our model when retraining with its own pseudolabels incurring no manual annotation cost.
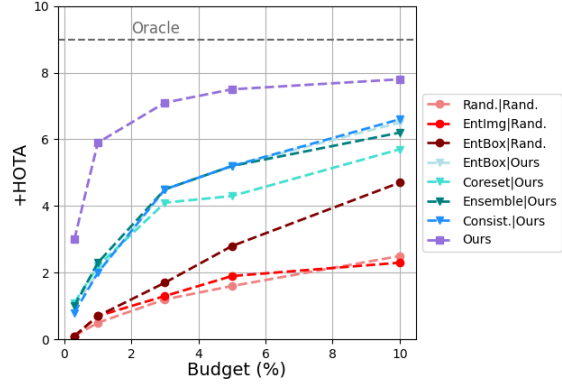
Figure 5: Impact of various active learning strategies on the quality of labeling.

Active learning on graph hierarchies. We now test the effectiveness of our proposed active learning labeling on graph hierarchies, and show a comprehensive comparison with several baselines in Fig. 5. We directly measure the accuracy of the output labels given an increasing percentage of manual annotations selected by different acquisition functions on MOT17. The vertical axis shows the increase in HOTA over the baseline performance, *i.e.*, the performance without active learning. *Oracle* is our upper bound and shows the performance if all possible labels are set to their correct value. Every method has two different labeling schemes: one for detections (nodes), and one for associations (edges) as explained in Section 3.3. In addition to our method, we explore active learning strategies commonly used in object detection literature, including Coreset [58], Ensemble [5], and Consistency over augmented inputs [21]. The naming convention denotes the active learning strategy for detection in the first part and the strategy for association in the second part. For instance, 'EntImg|Rand' indicates image level entropy-based selection for detection and random selection for association.

The methods shown in red shades and circle markers compare three detection labeling methods while keeping association label selection random. Box-level labeling is far superior to commonly-used image-level labeling, or random labeling. This highlights the potential of performing active learning at the object level rather than the frame level. The methods in blue shades and triangle markers all use our proposed active learning selection on graph hierarchies for the association, and compare several detection selection schemes. The Coreset [58], which prioritizes diversity of the data over uncertainty, emerges as the poorest performing approach. This observation underlines that, when accurate predictions are available, an uncertainty-based approach proves to be a more effective complement to pseudo-labels. Box-level entropy, ensembles, and consistency all perform similarly. Finally, our method, with both detection and association label selection on graph hierarchies. Its performance is far superior to all compared baselines, staying at only 1.2 points from Oracle performance. Notably, we also unify the AL strategy for detection and association with our approach.

Model	Labels	Pseudolabels	Active Learn.	Ann. Effort ↓	HOTA ↑	MOTA ↑	IDF1 ↑
ByteTrack [78]	Ground Truth			100%	**52.6**	60.4	65.7
ByteTrack [78]	**SP**A**M**	✗	✗	✗	49.6	57.8	63.1
ByteTrack [78]	**SP**A**M**	✓	✗	✗	50.5	59.5	64.5
ByteTrack [78]	**SP**A**M**	✓	✓	3.3%	52.5	**61.8**	**66.2**

Table 4: Effect of (i) **S**ynthetic training, (ii) Retraining our model with our **P**seudo-labels, and (iii) **A**ctive learning-based acquisition and annotation of a small percentage of real data, on our label quality on the MOT17 validation set.

Method	Labels	Ann. Effort ↓	HOTA ↑	MOTA ↑	IDF1 ↑
MOT17 Test					
ByteTrack [78]	GT	100%	50.6	60.4	61.1
ByteTrack [78]	SPAM	3.3%	**51.6**	**64.0**	**63.0**
GHOST [57]	GT	100%	49.5	58.0	59.0
GHOST [57]	SPAM	3.3%	**51.3**	**61.9**	**62.1**
MOT20 Test					
ByteTrack [78]	GT	100%	**48.7**	60.5	60.0
ByteTrack [78]	SPAM	3.3%	47.9	57.6	61.4
ByteTrack [78]	SPAM	10%	48.4	**61.0**	**62.8**
GHOST [57]	GT	100%	**48.3**	59.7	60.2
GHOST [57]	SPAM	3.3%	47.0	58.2	60.7
GHOST [57]	SPAM	10%	46.9	59.0	**60.8**
DanceTrack Test					
ByteTrack [78]	GT	100%	41.0	79.1	46.0
ByteTrack [78]	SPAM	3.3%	39.5	76.4	45.0
ByteTrack [78]	SPAM	20%	**41.3**	**80.9**	**48.1**
GHOST [57]	GT	100%	**45.8**	80.6	46.3
GHOST [57]	SPAM	3.3%	41.0	76.3	44.8
GHOST [57]	SPAM	20%	43.9	**81.3**	**47.7**

Table 5: Evaluation of SPAM labels on MOT17, MOT20, and DanceTrack test sets with varying annotation budgets by training ByteTrack [78] and GHOST [57] using annotations generated by our method.

4.6 Putting Everything Together: SPAM Labels in Action

Training trackers with our labels. In this experiment, we ablate the effects of our SPAM pipeline on the actual usecase of training a model with our provided labels. We ablate label quality under the following settings: (i) Synthetic training, (ii) Retraining our models on our Pseudo-labels, (iii) Active learning based acquisiton and annotation of a small percentage of real data. We train state-of-the-art online tracker ByteTrack [78] from scratch with the aforementioned sets of labels and report its performance on the MOT17 validation set in Table 4. The goal performance we want to reach is the one achieve by the model trained on groundtruth annotations (row 1). Row 2 shows our model trained only on synthetic data, which is 3 HOTA points below our goal. Using pseudo-label based training of our model increases performance by almost a HOTA point, and the ground truth level performance is achieved by also using our active learning method to select which samples to label. By annotating only 3.3% of the data, we can achieve the same results as the model trained on 100% of the ground truth data. This shows the potential of our SPAM in reducing annotation effort for multi-object tracking datasets.

How much data do we really need to label? In this final experiment, our goal is to assess the impact of our entire pipeline by answering the question: How much data do we actually need to label in order to achieve the same performance as a model trained on the entire groundtruth training dataset? For that, we choose two well-established trackers, ByteTrack [78] and GHOST [57] due to their online inference strategy and versatility, and train them from scratch using our SPAM pipeline. We report their performance on MOT17, MOT20, and DanceTrack test sets in Table 5. We first confirm that our results on the validation set also generalize to the test set, and that we can reach GT performance with 3.3% of annotation effort on MOT17. Note that for MOT17 our annotations yield a slight improvement over GT, which can be attributed to observed annotation noise in the original GT, particularly in moving sequences. Further details can be found in the supplementary material. MOT20 and DanceTrack are known to be more challenging datasets [60], which is confirmed by the

fact that the trackers still miss a few points when trained on only 3.3% of manually annotated data. Given that we cannot do a full analysis of different percentages because we are working on the test sets, we choose two percentages of manually annotated data on which to train our trackers: 10% for MOT20, and 20% for DanceTrack. In both cases, we achieve on-par performance with the trackers trained on full groundtruth data. This clearly shows the potential in properly leveraging synthetic data, pseudo-labels from a strong model, and active learning on graph hierarchies in order to label multi-object tracking data, where we can save 96.7% of the annotation effort on MOT17, 90% of the annotation effort on MOT20 and 80% of the annotation effort on DanceTrack.

5   Conclusion

We have introduced SPAM, a labeling engine to efficiently obtain trajectory annotations from video. We have shown that SPAM is able to produce annotations that yield state-of-the-art tracking performance with a significantly reduced human annotation effort. Through our extensive experiments we have shown that (i) our synthetic pretraining + pseudo-label finetuning recipe yields a strong, annotation-free, labeler; (ii) our combination of active learning + pseudo-labeling for annotation generation yields high quality and efficient annotations; and (iii) that graphs are an excellent tool to provide comprehensive modelling for trajectory annotation. We believe that SPAM paves the way to make large-scale cheap annotation of tracking datasets a reality, which will in turn power the next generation of data-hungry tracking algorithms.

References

[1] D. Acuna, H. Ling, A. Kar, and S. Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, 2018.

[2] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[3] A. Andriyenko and K. Schindler. Multi-target tracking by continuous energy minimization. In *CVPR*, pages 1265–1272, 2011.

[4] F. Bastani, S. He, and S. Madden. Self-supervised multi-object tracking with cross-input consistency. *Advances in Neural Information Processing Systems*, 34:13695–13706, 2021.

[5] W. H. Beluch, T. Genewein, A. Nürnberger, and J. M. Köhler. The power of ensembles for active learning in image classification. In *CVPR*, pages 9368–9377, 2018.

[6] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua. Multiple object tracking using k-shortest paths optimization. *IEEE TPAMI*, 33(9):1806–1819, 2011.

[7] P. Bergmann, T. Meinhardt, and L. Leal-Taixe. Tracking without bells and whistles. In *ICCV*, pages 941–951, 2019.

[8] G. Brasó, O. Cetintas, and L. Leal-Taixé. Multi-object tracking and segmentation via neural message passing. *IJCV*, 130(12):3035–3053, 2022.

[9] G. Braso and L. Leal-Taixe. Learning a neural solver for multiple object tracking. In *CVPR*, 2020.

[10] J. Cai, M. Xu, W. Li, Y. Xiong, W. Xia, Z. Tu, and S. Soatto. Memot: Multi-object tracking with memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8090–8100, 2022.

[11] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.

[12] O. Cetintas, G. Brasó, and L. Leal-Taixé. Unifying short and long-term tracking with graph hierarchies. In *CVPR*, pages 22877–22887, June 2023.

[13] L. Chen, H. Ai, Z. Zhuang, and C. Shang. Real-time multiple people tracking with deeply learned candidate selection and person re-identification. In *2018 IEEE international conference on multimedia and expo (ICME)*, pages 1–6. IEEE, 2018.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *IEEE Int. Conf. Mach. Learn.*, pages 1597–1607. PMLR, 2020.

[15] P. Chu and H. Ling. Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking. In *ICCV*, October 2019.

[16] K. Dai, J. Zhao, L. Wang, D. Wang, J. Li, H. Lu, X. Qian, and X. Yang. Video annotation for visual tracking via selection and refinement. In *ICCV*, 2021.

[17] Z. Dai, B. Cai, Y. Lin, and J. Chen. Up-detr: Unsupervised pre-training for object detection with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1601–1610, June 2021.

[18] P. Dendorfer, A. Osep, A. Milan, K. Schindler, D. Cremers, I. Reid, S. Roth, and L. Leal-Taixé. Motchallenge: A benchmark for single-camera multiple target tracking. *International Journal of Computer Vision*, 129(4):845–881, 2021.

[19] P. Dendorfer, H. Rezatofighi, A. Milan, J. Q. Shi, D. Cremers, I. D. Reid, S. Roth, K. Schindler, and L. Leal-Taix'e. Mot20: A benchmark for multi object tracking in crowded scenes. *ArXiv*, abs/2003.09003, 2020.

[20] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: A benchmark. In *CVPR*, pages 304–311. IEEE, 2009.

[21] I. Elezi, Z. Yu, A. Anandkumar, L. Leal-Taixe, and J. M. Alvarez. Not all labels are equal: Rationalizing the labeling costs for training object detection. In *CVPR*, pages 14492–14501, 2022.

[22] A. Ess, B. Leibe, K. Schindler, and L. Van Gool. A mobile vision system for robust multi-person tracking. In *CVPR*, pages 1–8, 2008.

[23] M. Fabbri, G. Brasó, G. Maugeri, O. Cetintas, R. Gasparini, A. Ošep, S. Calderara, L. Leal-Taixé, and R. Cucchiara. Motsynth: How can synthetic data help pedestrian detection and tracking? In *ICCV*, pages 10849–10859, October 2021.

[24] M. Fabbri, F. Lanzi, S. Calderara, A. Palazzi, R. Vezzani, and R. Cucchiara. Learning to detect and track visible and occluded body joints in a virtual world. In *ECCV*, 2018.

[25] C. Feichtenhofer, A. Pinz, and A. Zisserman. Detect to track and track to detect. In *ICCV*, Oct 2017.

[26] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, pages 4340–4349, 2016.

[27] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[29] H.-N. Hu, Q.-Z. Cai, D. Wang, J. Lin, M. Sun, P. Krähenbühl, T. Darrell, and F. Yu. Joint monocular 3d vehicle detection and tracking. In *ICCV*, 2019.

[30] A. Jabri, A. Owens, and A. Efros. Space-time correspondence as a contrastive random walk. *Advances in neural information processing systems*, 33:19545–19560, 2020.

[31] H. Jiang, S. Fels, and J. J. Little. A linear programming approach for multiple object tracking. In *CVPR*, pages 1–8, 2007.

[32] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, M. Boonstra, V. Korzhova, and J. Zhang. Framework for performance evaluation for face, text and vehicle detection and tracking in video: data, metrics, and protocol. *IEEE TPAMI*, 2009.

[33] P. Kaul, W. Xie, and A. Zisserman. Label, verify, correct: A simple few-shot object detection method. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[34] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[35] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *ICCV*, 2023.

[36] S. Lan, X. Yang, Z. Yu, Z. Wu, J. M. Alvarez, and A. Anandkumar. Vision transformers are good mask auto-labelers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2023.

[37] L. Leal-Taixe, C. Canton-Ferrer, and K. Schindler. Learning by tracking: Siamese cnn for robust target association. In *CVPRW*, June 2016.

[38] L. Leal-Taixe, M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese. Learning an image-based motion context for multiple people tracking. In *CVPR*, June 2014.

[39] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *Int. Conf. Comput. Vis. Worksh.*, pages 120–127, 2011.

[40] R. Li, B. Zhang, J. Liu, W. Liu, J. Zhao, and Z. Teng. Heterogeneous diversity driven active learning for multi-object tracking. In *ICCV*, pages 9932–9941, 2023.

[41] R. Li, B. Zhang, J. Liu, W. Liu, J. Zhao, and Z. Teng. Heterogeneous diversity driven active learning for multi-object tracking. In *ICCV*, 2023.

[42] M. Litrico, A. Del Bue, and P. Morerio. Guiding pseudo-labels with uncertainty estimation for source-free unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[43] K. Liu, S. Jin, Z. Fu, Z. Chen, R. Jiang, and J. Ye. Uncertainty-aware unsupervised multi-object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9996–10005, 2023.

[44] J. Luiten, A. Osep, P. Dendorfer, P. Torr, A. Geiger, L. Leal-Taixé, and B. Leibe. Hota: A higher order metric for evaluating multi-object tracking. *IJCV*, 129(2):548–578, 2021.

[45] S. Manen, M. Gygli, D. Dai, and L. Van Gool. Pathtrack: Fast trajectory annotation with path supervision. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 290–299, 2017.

[46] T. Meinhardt, A. Kirillov, L. Leal-Taixe, and C. Feichtenhofer. Trackformer: Multi-object tracking with transformers. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2022.

[47] S. Meng, D. Shao, J. Guo, and S. Gao. Tracking without label: Unsupervised multiple object tracking via contrastive similarity learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16264–16273, 2023.

[48] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv preprint arXiv:1603.00831*, 2016.

[49] J. Pang, L. Qiu, X. Li, H. Chen, Q. Li, T. Darrell, and F. Yu. Quasi-dense similarity learning for multiple object tracking. In *CVPR*, pages 164–173, 2021.

[50] S. Pellegrini, A. Ess, K. Schindler, and L. van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, pages 261–268, 2009.

[51] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes. Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR*, pages 1201–1208, 2011.

[52] Z. Qin, S. Zhou, L. Wang, J. Duan, G. Hua, and W. Tang. Motiontrack: Learning robust short-term and long-term motions for multi-object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17939–17948, 2023.

[53] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi. Performance measures and a data set for multi-target, multi-camera tracking. In *Eur. Conf. Comput. Vis. Worksh.*, pages 17–35. Springer, 2016.

[54] E. Ristani and C. Tomasi. Features for multi-target multi-camera tracking and re-identification. In *CVPR*, June 2018.

[55] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Learning social etiquette: Human trajectory understanding in crowded scenes. In *ECCV*, 2016.

[56] P. Scovanner and M. F. Tappen. Learning pedestrian dynamics from the real world. In *2009 IEEE 12th International Conference on Computer Vision*, pages 381–388. IEEE, 2009.

[57] J. Seidenschwarz, G. Brasó, V. C. Serrano, I. Elezi, and L. Leal-Taixé. Simple cues lead to a strong multi-object tracker. In *CVPR*, pages 13813–13823, 2023.

[58] O. Sener and S. Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018.

[59] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[60] P. Sun, J. Cao, Y. Jiang, Z. Yuan, S. Bai, K. Kitani, and P. Luo. Dancetrack: Multi-object tracking in uniform appearance and diverse motion. In *CVPR*, 2022.

[61] S. Sun, N. Akhtar, X. Song, H. Song, A. Mian, and M. Shah. Simultaneous detection and tracking with motion modelling for multiple object tracking. In *ECCV*, pages 626–643, 2020.

[62] P. Voigtlaender, M. Krause, A. Osep, J. Luiten, B. B. G. Sekar, A. Geiger, and B. Leibe. Mots: Multi-object tracking and segmentation. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition*, pages 7942–7951, 2019.

[63] P. Voigtlaender, L. Luo, C. Yuan, Y. Jiang, and B. Leibe. Reducing the annotation effort for video object segmentation datasets. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3060–3069, 2021.

[64] C. Vondrick and D. Ramanan. Video annotation and tracking with active learning. *NeurIPS*, 24, 2011.

[65] C. Vondrick, D. Ramanan, and D. Patterson. Efficiently scaling up video annotation with crowdsourced marketplaces. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 610–623. Springer, 2010.

[66] C. Vondrick, A. Shrivastava, A. Fathi, S. Guadarrama, and K. Murphy. Tracking emerges by colorizing videos. In *Proceedings of the European conference on computer vision (ECCV)*, pages 391–408, 2018.

[67] X. Wang, A. Jabri, and A. A. Efros. Learning correspondence from the cycle-consistency of time. In *CVPR*, 2019.

[68] Y. Wang, K. Kitani, and X. Weng. Joint object detection and multi-object tracking with graph neural networks. In *IEEE Int. Conf. Robotics and Autom.*, pages 13708–13715, 2021.

[69] Z. Wang, L. Zheng, Y. Liu, and S. Wang. Towards real-time multi-object tracking. *The European Conference on Computer Vision (ECCV)*, 2020.

[70] F. Wei, Y. Gao, Z. Wu, H. Hu, and S. Lin. Aligning pretraining for detection via object-level contrastive learning. In *Advances in Neural Information Processing Systems*, 2021.

[71] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang. Joint detection and identification feature learning for person search. In *CVPR*, pages 3415–3424, 2017.

[72] K. Yamaguchi, A. C. Berg, L. E. Ortiz, and T. L. Berg. Who are you with and where are you going? In *CVPR*, pages 1345–1352, 2011.

[73] S. You, H. Yao, B.-K. Bao, and C. Xu. Utm: A unified multiple object tracking model with identity-aware feature enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21876–21886, 2023.

[74] D. Yuan, X. Chang, Q. Liu, Y. Yang, D. Wang, M. Shu, Z. He, and G. Shi. Active learning for deep visual tracking. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[75] F. Zeng, B. Dong, Y. Zhang, T. Wang, X. Zhang, and Y. Wei. Motr: End-to-end multiple-object tracking with transformer. In *European Conference on Computer Vision*, pages 659–675. Springer, 2022.

[76] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.

[77] S. Zhang, R. Benenson, and B. Schiele. Citypersons: A diverse dataset for pedestrian detection. In *CVPR*, pages 3213–3221, 2017.

[78] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang. Bytetrack: Multi-object tracking by associating every detection box. In *ECCV*, pages 1–21. Springer, 2022.

[79] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu. Fairmot: On the fairness of detection and re-identification in multiple object tracking. *IJCV*, 129(11):3069–3087, 2021.

[80] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian. Person re-identification in the wild. In *CVPR*, pages 1367–1376, 2017.

[81] X. Zhou, V. Koltun, and P. Krähenbühl. Tracking objects as points. In *ECCV*, pages 474–490. Springer, 2020.

Appendix

Abstract

This supplementary material offers additional details (Appendix A) and experimental results (Appendix B) of our SPAM video data label engine. To this end, we provide further information on the architecture of our graph-based model (Appendix A.1) and its application for hierarchical labeling (Appendix A.2). In Appendix A.3, we complete the implementation details of the main paper. The experimental results in Appendix B.1 demonstrate the superiority of our approach to a common frame-based labeling method. Finally, we compare labeling performance for static and moving cameras (Appendix B).

A Additional Details about SPAM

A.1 GNN Model

Our hierarchical graph-neural network (GNN) utilizes the time-aware message passing framework introduced in [9]. We use a hierarchy of 10 GNNs operating over a video clip of 512 frames. At the first level, we employ a GNN_{node}, which considers a temporal window of 10 frames for each node. Subsequent levels, denoted as GNN_{edge}, cover 2, 4, ..., up to 512 frames. We follow a simple and unified approach for all hierarchy levels: GNN_{node} and GNN_{edge} levels share the same input features and network architecture (as presented in Section 3.2 of the main paper). Moreover, GNN_{edge} levels also share learnable weights across hierarchy levels. Overall, we rely on a lightweight GNN hierarchy with only approximately 65K parameters running at 20 frames per second (FPS).

A.2 Active Hierarchical Labeling

To fully utilize our hierarchical graph framework, we distribute our annotation budget B across L hierarchical levels as $B_1, \ldots, B_L$, with $B_1 + \cdots + B_L = B$. Prioritizing deeper levels enables a more effective allocation of the annotation budget. Intuitively, in deeper hierarchy levels, nodes represent tracklets rather than individual detections. We propagate annotator decisions for entire clusters rather than individual detections, resolving multiple uncertainties with a single annotation. This makes the annotation pipeline more efficient compared to working at the detection level. For example for MOT17 [48], 50% of the budget B is allocated to the last three levels, 30% to the previous three levels, and the remaining 20% to earlier levels. For MOT17, we do not allocate a budget for refining bounding boxes since our experiments revealed that the bounding boxes generated by our synthetic detector are already well localized. However, for the challenging DanceTrack [60] dataset, we allocate a portion (approximately 30%) of our annotation budget B for refining bounding boxes.
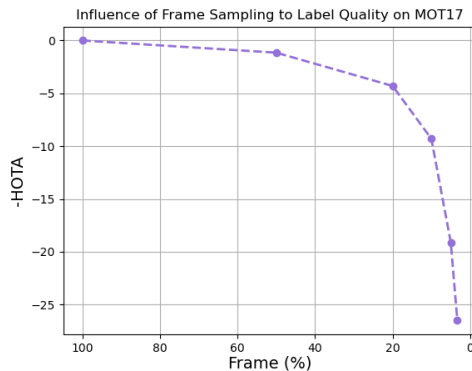
Figure 6: Annotation quality (reported as relative HOTA drop) on MOT17 by uniformly labeling different percentages of frames, and then interpolating the ground truth boxes.
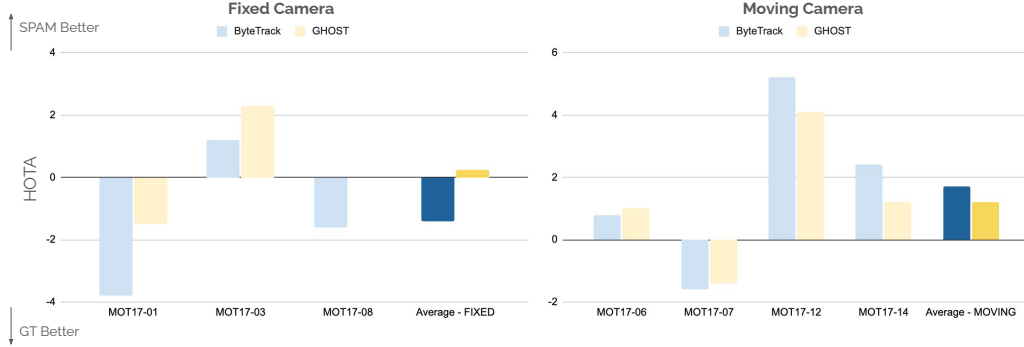
Figure 7: Comparison of MOT17 results for static and moving camera sequences.

A.3 Implementation Details

Architecture and training. We train our YOLOX [27] detector on MOTSynth for 170 epochs following the training parameters in [78]. We keep augmentations on throughout the training. We train our GNN hierarchy on MOTSynth jointly for 10 epochs, with a learning rate of $3 * 10^{-4}$ and weight decay of 10^{-4} with batches of 4 graphs. We use a focal loss with $\gamma = 1$ and the Adam optimizer [34].

Experimental setup. After obtaining labels with SPAM, we train two well-established online trackers ByteTrack [78] and GHOST [57] to evaluate our label quality in Section 4.6 of the main paper. Our main goal is to compare our label quality with ground truth label quality, therefore, we start trainings from scratch and only train on the dataset of interest (200 epochs for MOT17, 150 epochs for MOT20 and 100 epochs for DanceTrack). During both training and testing, we follow their default training and testing parameters [78, 57].

B Additional Experiments

B.1 Frame-based Labeling and FPS

The most related work to active learning in the MOT domain is [41], a frame-based labeling method. Motivated by the redundancy of nearby video frames, they propose to label only a portion of frames. However, their method requires a large labeling budget (50% of the frames) to achieve ground-truth level performance. Furthermore, the authors [41] report that at lower data regimes (20% to 40%) their method is outperformed by a simple uniform sampling of frames.

To demonstrate the inefficiency of such a frame-based labeling method, we show that for high frame rates, labeling such a large portion of the frames will trivially lead to ground-truth performance. This is attributed to the redundancy of data in MOT17 sequences, which are annotated with high frames per second (FPS) of up to 30. To this end, we examine the annotation quality on MOT17 by uniformly labeling different percentages of frames, *i.e.*, reducing the FPS, and then interpolating the ground truth boxes. We report relative HOTA drops compared to labeling all frames in Fig. 6. Notably, we observe that a naive solution of labeling 50% of the frames (every other frame) and interpolating the bounding boxes yields almost perfect labels (1 HOTA drop, leading to 99 HOTA). However, a significant performance drop is observed for lower data regimes, i.e., using lower than 20% of frames equivalent to approximately 6 FPS for MOT17. This highlights that labeling becomes notably more challenging for low data regimes in which [41] fails to operate.

In contrast to frame-based labeling, SPAM performs active learning for multiple objects across all video frames, utilizing the uncertainty measures from our graph-based model. This allows us to invest expensive annotation efforts on *individual tracks* instead of labeling *entire frames* in a brute-force manner. Consequently, we can reach ground-truth level performance with a minimal annotation budget of only 3.3% for MOT17.

B.2 Label Noise for Moving Cameras

In our experiments detailed in Section 4.6 of the main paper, we found that on MOT17, our annotations with a budget of 3.3% yield slightly better performance compared to the ground truth. We attribute this observation to annotation noise in the original ground truth, particularly in sequences with moving cameras. To illustrate this, we provide detailed results on MOT17 for each sequence with ByteTrack and GHOST in Fig. 7. We compare relative HOTA scores between trainings on ground truth and SPAM labels. Positive scores indicate superior performance with SPAM labels, while negative scores indicate that ground truth labels result in better performance. Overall, with a 3.3% budget, ground truth exhibits better performance on static camera sequences, while SPAM outperforms ground truth labels on moving camera sequences. We attribute this to additional noise in ground truth labels on moving sequences caused by the interpolation-based labeling strategy reported in [18].