

# Writing a Paper & Research Career Paths

CS 197 | Stanford University | Michael Bernstein

# Today's goals

We have a bunch of things we tried, some of them worked, some of them didn't — how do we write a paper about this?

Introducing the concept of **model papers** and how to use them

What happens if I keep doing research at Stanford? And after?

# Writing A Paper

# Scene Graph Prediction with Limited Labels

Shen S. Chen, Paroma Varma, Ranjay Krishna, Michael Bernstein, Christopher Ré, Li Fei-Fei  
Stanford University

{vincentsc, paroma, ranjaykrishna, msb, chrismre, feifeili}@cs.stanford.edu

## Abstract

Knowledge bases such as Visual Genome power applications in computer vision, including visual question answering and captioning, but suffer from sparse relationships. All scene graph models to date rely on training on a small set of visual relationships with thousands of training labels each. Hiring humans is expensive, and using textual knowledge bases in methods are incompatible with visual data. In this paper, we introduce a semi-supervised method that automatically generates probabilistic relationship labels to train any scene graph model. We analyze the complexity of relationships and show that our method outperforms all baselines on scene graph prediction by 5.16 recall@100 on PREDCLS. In our limited label setting, we define a metric for relationships that serves as an indicator of their complexity under which our method outperforms transfer learning, the de-facto approach for limited labels.

## Introduction

Effort to formalize a structured representation for Visual Genome [27] defined scene graphs, a formalism similar to those widely used to represent knowledge [13, 18, 56]. Scene graphs encode objects (e.g. bike) as nodes connected via pairwise relationships (e.g. riding) as edges. This formalization has led to state-of-the-art models in image captioning [3], image question answering [24], relationship classification [26] and image generation [23]. However, existing scene graph models ignore more than 98% of relationships that do not have sufficient labeled instances (see Figure 2) and instead focus on modeling the

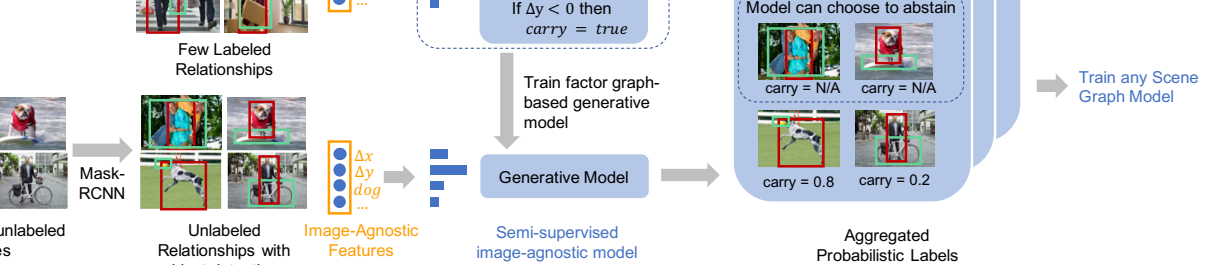


Figure 1. Our semi-supervised method automatically generates probabilistic relationship labels to train any scene graph model.

few relationships that have thousands of labels [31, 49, 54]. Hiring more human workers is an ineffective solution to labeling relationships because image annotation is so tedious that seemingly obvious labels are left unannotated. To complement human annotators, traditional text-based knowledge completion tasks have leveraged numerous semi-supervised or distant supervision approaches [6, 7, 17, 34]. These methods find syntactical or lexical patterns from a small labeled set to extract missing relationships from a large unlabeled set. In text, pattern-based methods are successful, as relationships in text are usually document-agnostic (e.g. <Tokyo is capital of - Japan>). Visual relationships are often incidental: they depend on the contents of the particular image they appear in. Therefore, methods that rely on external knowledge or on patterns over concepts (e.g. most instances of dog next to frisbee are playing with it) do not generalize well. The inability to utilize the progress in text-based methods necessitates specialized methods for visual knowledge.

In this paper, we automatically generate missing relationship labels using a small, labeled dataset and use these generated labels to train downstream scene graph models (see Figure 1). We begin by exploring how to define image-agnostic features for relationships so they follow patterns across images. For example, eat usually consists of one object consuming another object smaller than itself, whereas look often consists of common objects: phone, laptop, or window (see Figure 3). These rules are not dependent on raw pixel values; they can be derived from image-agnostic features like object categories and relative spatial positions between objects in a relationship. While such rules are simple, their capacity to provide supervision for unannotated relationships has been unexplored. While image-agnostic

features can characterize some visual relationships very well, they might fail to capture complex relationships with high variance. To quantify the efficacy of our image-agnostic features, we define “subtypes” that measure spatial and categorical complexity (Section 3).

Based on our analysis, we propose a semi-supervised approach that leverages image-agnostic features to label missing relationships using as few as 10 labeled instances of each relationship. We learn simple heuristics over these features and assign probabilistic labels to the unlabeled images using a generative model [39, 46]. We evaluate our method’s labeling efficacy using the completely-labeled VRD dataset [31] and find that it achieves an F1 score of 57.66, which is 11.84 points higher than other standard semi-supervised methods like label propagation [57]. To demonstrate the utility of our generated labels, we train a state-of-the-art scene graph model using [54] (see Figure 6) and modify its loss function to support probabilistic labels. Our approach achieves 47.53 recall@100<sup>1</sup> for predicate classification on Visual Genome, improving over the same model trained using only labeled instances by 40.97 points. For scene graph detection, our approach achieves within 8.65 recall@100 of the same model trained on the original Visual Genome dataset with 108× more labeled data. We end by comparing our approach to transfer learning, the de-facto choice for learning from limited labels. We find that our approach improves by 5.16 recall@100 for predicate classification, especially for relationships with high complexity, as it generalizes well to unlabeled subtypes.

Our contributions are three-fold. (1) We introduce the first method to complete visual knowledge bases by finding missing visual relationships (Section 5.1). (2) We show the utility of our generated labels in training existing scene graph prediction models (Section 5.2). (3) We introduce a metric to characterize the complexity of visual relationships and show it is a strong indicator ( $R^2 = 0.778$ ) for our semi-supervised method’s improvements over transfer learning (Section 5.3).

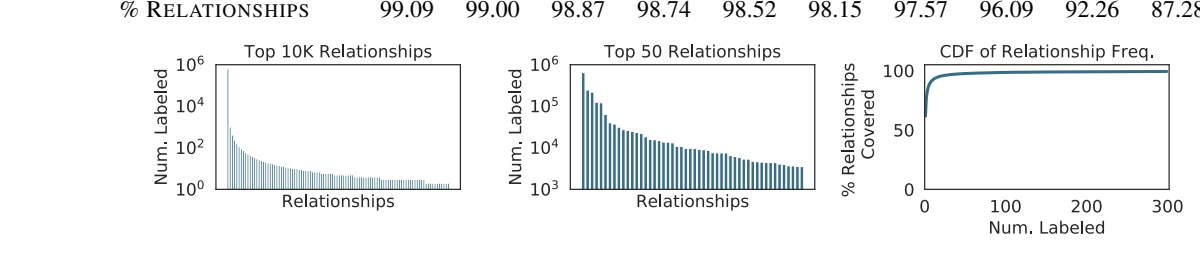


Figure 2. Visual relationships have a long tail (left) of infrequent relationships. Current models [49, 54] only focus on the top 50 relationships (middle) in the Visual Genome dataset, which all have thousands of labeled instances. This ignores more than 98% of the relationships with few labeled instances (right, top/table).

features can characterize some visual relationships very well, they might fail to capture complex relationships with high variance. To quantify the efficacy of our image-agnostic features, we define “subtypes” that measure spatial and categorical complexity (Section 3).

Based on our analysis, we propose a semi-supervised approach that leverages image-agnostic features to label missing relationships using as few as 10 labeled instances of each relationship. We learn simple heuristics over these features and assign probabilistic labels to the unlabeled images using a generative model [39, 46]. We evaluate our method’s labeling efficacy using the completely-labeled VRD dataset [31] and find that it achieves an F1 score of 57.66, which is 11.84 points higher than other standard semi-supervised methods like label propagation [57]. To demonstrate the utility of our generated labels, we train a state-of-the-art scene graph model using [54] (see Figure 6) and modify its loss function to support probabilistic labels. Our approach achieves 47.53 recall@100<sup>1</sup> for predicate classification on Visual Genome, improving over the same model trained using only labeled instances by 40.97 points. For scene graph detection, our approach achieves within 8.65 recall@100 of the same model trained on the original Visual Genome dataset with 108× more labeled data. We end by comparing our approach to transfer learning, the de-facto choice for learning from limited labels. We find that our approach improves by 5.16 recall@100 for predicate classification, especially for relationships with high complexity, as it generalizes well to unlabeled subtypes.

Our contributions are three-fold. (1) We introduce the first method to complete visual knowledge bases by finding missing visual relationships (Section 5.1). (2) We show the utility of our generated labels in training existing scene graph prediction models (Section 5.2). (3) We introduce a metric to characterize the complexity of visual relationships and show it is a strong indicator ( $R^2 = 0.778$ ) for our semi-supervised method’s improvements over transfer learning (Section 5.3).

Our contributions are three-fold. (1) We introduce the first method to complete visual knowledge bases by finding missing visual relationships (Section 5.1). (2) We show the utility of our generated labels in training existing scene graph prediction models (Section 5.2). (3) We introduce a metric to characterize the complexity of visual relationships and show it is a strong indicator ( $R^2 = 0.778$ ) for our semi-supervised method’s improvements over transfer learning (Section 5.3).

Our contributions are three-fold. (1) We introduce the first method to complete visual knowledge bases by finding missing visual relationships (Section 5.1). (2) We show the utility of our generated labels in training existing scene graph prediction models (Section 5.2). (3) We introduce a metric to characterize the complexity of visual relationships and show it is a strong indicator ( $R^2 = 0.778$ ) for our semi-supervised method’s improvements over transfer learning (Section 5.3).

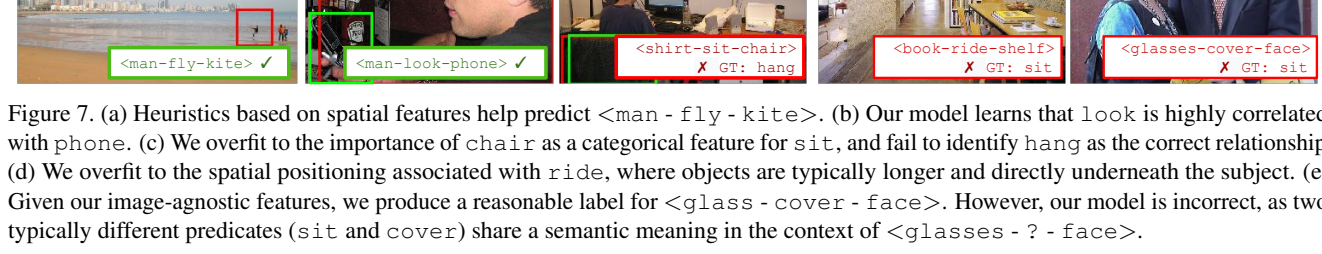


Figure 7. (a) Heuristics based on spatial features help predict <man-fly-kite>. (b) Our model learns that look is highly correlated with phone. (c) We overfit to the importance of chair as a categorical feature for sit, and fail to identify hang as the correct relationship. (d) We overfit to the spatial positioning associated with ride, where objects are typically longer and directly underneath the subject. (e) Given our image-agnostic features, we produce a reasonable label for <glass-cover-face>. However, our model is incorrect, as two typically different predicates (sit and cover) share a semantic meaning in the context of <glasses-?-face>.

that our semi-supervised method outperforms transfer learning, which has seen more data. Furthermore, we quantify when our method outperforms transfer learning using our metric for measuring relationship complexity (Section 3.3). Eliminating synonyms and supersets. Typically, past scene graph approaches have used 50 predicates from Visual Genome to study visual relationships. Unfortunately, past methods treat synonyms like lying on and lying on top of as distinct, which is not ideal. To make matters worse, some methods generate training labels that can then be used to train scene graph models. ORACLE [10], a state-of-the-art scene graph model, uses 108 unique predicates from Visual Genome, which amounts to a large number of labels. In this paper, we compare to a more compact set of labels. In this section, we eliminate all synonyms and supersets, resulting in 20 unique predicates. In the Supplementary Material we include a list of these predicates and report our method’s performance on all 50 predicates.

Dataset. We use two standard datasets, VRD [31] and Visual Genome [27], to evaluate on tasks related to visual relationships or scene graphs. Each scene graph contains objects localized as bounding boxes in the image along with pairwise relationships connecting them, categorized as action (e.g., carry), possessive (e.g., wear), spatial (e.g., above), or comparative (e.g., taller than) descriptors.

## 2. Related work

Textual knowledge bases were originally hand-curated by experts to structure facts [4, 5, 44] (e.g. <Tokyo-capital of - Japan>). To scale dataset curation efforts, recent approaches mine knowledge from the web [9] or hire non-expert annotators to manually curate knowledge [5, 47]. In semi-supervised solutions, a small amount of labeled text is used to extract and exploit patterns in unlabeled sentences [2, 21, 33–35, 37]. Unfortunately, such approaches cannot be directly applied to visual relationships; textual relations can often be captured by external knowledge or patterns, while visual relationships are often local to an image.

Visual relationships have been studied as spatial priors [14, 16], co-occurrences [51], language statistics [28, 31, 53], and within entity contexts [29]. Scene graph prediction models have dealt with the difficulty of learning from incomplete knowledge, as recent methods utilize statistical motifs [54] or object-relationship dependencies [30, 49, 50, 55]. All these methods limit their inference to the top 50 most frequently occurring predicate categories and ignore those without enough labeled examples (Figure 2).

The de-facto solution for limited label problems is transfer learning [15, 52], which requires that the source domain used for pre-training follows a similar distribution as the target domain. In our setting, the source domain is a dataset of frequently-labeled relationships with thousands of examples [30, 49, 50, 55], and the target domain is a set of limited label relationships. Despite similar objects in source and target domains, we find that transfer learning has difficulty generalizing to new relationships. Our method does not rely on availability of a larger, labeled set of relationships; instead, we use a small labeled set to annotate the unlabeled set of images.

To address the issue of gathering enough training labels for machine learning models, data programming has emerged as a popular paradigm. This approach learns to model imperfect labeling sources in order to assign training labels to unlabeled data. Imperfect labeling sources can come from crowdsourcing [10], user-defined heuristics [8, 43], multi-instance learning [22, 40], and distant su-

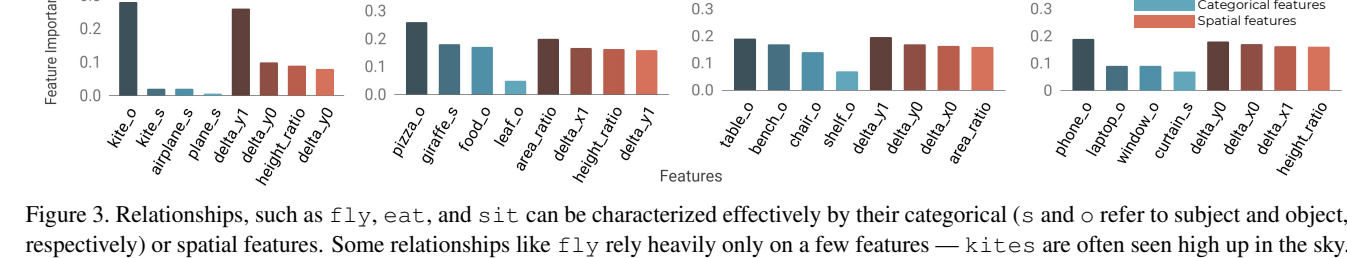


Figure 3. Relationships, such as fly, eat, and sit can be characterized effectively by their categorical (s and o refer to subject and object, respectively) or spatial features. Some relationships like fly rely heavily only on a few features — kites are often seen high up in the sky.

pervision [12, 32]. Often, these imperfect labeling sources take advantage of domain expertise from the user. In our case, imperfect labeling sources are automatically generated heuristics, which we aggregate to assign a final probabilistic label to every pair of object proposals.

## 3. Analyzing visual relationships

We define the formal terminology used in the rest of the paper and introduce the image-agnostic features that our semi-supervised method relies on. Then, we seek quantitative insights into how visual relationships can be described by the properties between its objects. We ask (1) what image-agnostic features can characterize visual relationships? and (2) given limited labels, how well do our chosen features characterize the complexity of relationships? With these in mind, we motivate our model design to generate heuristics that do not overfit to the small amount of labeled data and assign accurate labels to the larger, unlabeled set.

### 3.1. Terminology

A scene graph is a multi-graph  $G$  that consists of objects  $o$  as nodes and relationships  $r$  as edges. Each object  $o_i = \{b_i, c_i\}$  consists of a bounding box  $b_i$  and its category  $c_i \in C$  where  $C$  is the set of all possible object categories (e.g. dog, frisbee). Relationships are denoted <subject-predicate-object> or <o-p-o’>.  $p \in P$  is a predicate, such as ride and eat. We assume that we have a small labeled set  $\{(o, p, o') \in D_p\}$  of annotated relationships for each predicate  $p$ . Usually, these datasets are on the order of a 10 examples or fewer. For our semi-supervised approach, we also assume that there exists a large set of images  $D_U$  without any labeled relationships.

### 3.2. Defining image-agnostic features

It has become common in computer vision to utilize pre-trained convolutional neural networks to extract features that represent objects and visual relationships [31, 49, 50]. Models trained with these features have proven robust in the presence of enough training labels but tend to overfit when presented with limited data (Section 5). Consequently, an open question arises: what other features can we utilize

to label relationships with limited data? Previous literature has combined deep learning features with extra information extracted from categorical object labels and relative spatial object locations [25, 31]. We define categorical features,  $\langle o, -, o' \rangle$ , as a concatenation of one-hot vectors of the subject  $o$  and object  $o'$ . We define spatial features as:

$$\frac{x-x'}{w}, \frac{y-y'}{h}, \frac{(y+h)-(y'+h')}{h}, \frac{(x+w)-(x'+w')}{w}, \frac{h'-w'}{w}, \frac{w'+h'}{w+h}$$

where  $b = [y, x, h, w]$  and  $b' = [y', x', h', w']$  are the top-left bounding box coordinates and their widths and heights.

To explore how well spatial and categorical features can describe different visual relationships, we train a simple decision tree model for each relationship. We plot the importances for the top 4 spatial and categorical features in Figure 3. Relationships like fly place high importance on the difference in y-coordinate between the subject and object, capturing a characteristic spatial pattern. Look, on the other hand, depends on the category of the objects (e.g. phone, laptop, window) and not on any spatial orientations.

### 3.3. Complexity of relationships

To understand the efficacy of image-agnostic features, we’d like to measure how well they can characterize the complexity of particular visual relationships. As seen in Figure 4, a visual relationship can be defined by a number of image-agnostic features (e.g. a person can ride a bike, or a dog can ride a surfboard). To systematically define this notion of complexity, we identify subtypes for each visual relationship. Each subtype captures one way that a relationship manifests in the dataset. For example, in Figure 4, ride contains one categorical subtype with <person-ride-bike> and another with <dog-ride-surfboard>. Similarly, a person might carry an object in different relative spatial orientations (e.g. on her head, to her side). As shown in Figure 5, visual relationships might have significantly different degrees of spatial and categorical complexity, and therefore a different number of subtypes for each. To compute spatial subtypes, we perform mean shift clustering [11] over the spatial features extracted from all the

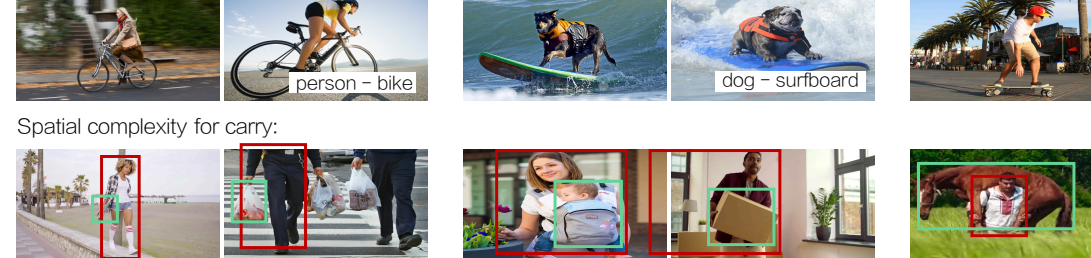


Figure 4. We define the number of subtypes of a relationship as a measure of its complexity. Subtypes can be categorized as <person-ride-bike> while another is <dog-ride-surfboard>. Categorical carry has a subtype with a small object carried to the side and another with a large object carried overhead.

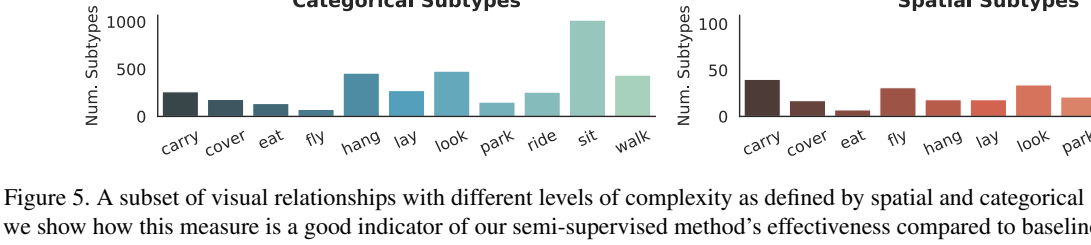


Figure 5. A subset of visual relationships with different levels of complexity as defined by spatial and categorical subtypes. We show how this measure is a good indicator of our semi-supervised method’s effectiveness compared to baseline

relationships in Visual Genome. To compute the categorical subtypes, we count the number of unique object categories associated with a relationship.

With access to 10 or fewer labeled instances for these visual relationships, it is impossible to capture all the subtypes for given relationship and therefore difficult to learn a good representation for the relationship as a whole. Consequently, we turn to the rules extracted from image-agnostic features and use them to assign labels to the unlabeled data in order to capture a larger proportion of subtypes in each visual relationship. We posit that this will be advantageous over methods that only use the small labeled set to train a scene graph prediction model, especially for relationships with high complexity, or a large number of subtypes. In Section 5.3, we find a correlation between our definition of complexity and the performance of our method.

## 4. Approach

We aim to automatically generate labels for missing visual relationships that can be then used to train any downstream scene graph prediction model. We assume that in the long-tail of infrequent relationships, we have a small labeled set  $\{(o, p, o') \in D_p\}$  of annotated relationships for each predicate  $p$  (often, on the order of a 10 examples or less). As discussed in Section 3, we want to leverage image-agnostic features to learn rules that annotate unlabeled relationships.

Our approach assigns probabilistic labels to a set  $D_U$  of un-annotated images in three steps: (1) we extract image-agnostic features from the objects in the labeled  $D_p$  and

### Algorithm 1 Semi-supervised Alg.

- 1: INPUT:  $\{(o, p, o') \in D_p\} \forall p \in P$  — A small set of multi-class labels for predicates.
- 2: INPUT:  $\{(o, o') \in D_U\}$  — A large unlabeled set of objects but no relationship labels.
- 3: INPUT:  $f(\cdot, \cdot)$  — A function that extracts features from images.
- 4: INPUT:  $DT(\cdot)$  — A decision tree.
- 5: INPUT:  $G(\cdot)$  — A generative model that assigns labels to unlabeled data.
- 6: INPUT:  $train(\cdot)$  — Function used to train a scene graph model.
- 7: Extract features and labels,  $X_p, Y_p := \{(o, o') \in D_p\}$ .
- 8: Generate heuristics by fitting  $J$  decision trees.
- 9: Assign labels to  $(o, o') \in D_U$ ,  $\Lambda = DT_{pred}$ .
- 10: Learn generative model  $G(\Lambda)$  and assign prob.
- 11: Train scene graph model,  $SGM := train(D_p, D_U, \Lambda)$ .
- 12: OUTPUT:  $SGM(\cdot)$ .

from the object proposals extracted using Mask-RCNN [19] on unlabeled  $D_U$ , (2) we use the image-agnostic features, and a factor-graph based generative model to assign probabilistic labels to the unlabeled data. These probabilistic labels, along with any scene graph prediction model. We show the end-to-end results in Algorithm 1 and show the end-to-end results in Figure 6. Feature extraction: Our approach uses features defined in Section 3, which represent objects and category labels. The feature extraction process involves ground truth objects in  $D_p$  or from object proposals in  $D_U$  by running existing object detectors. Heuristic generation: We fit decision trees to learn image-agnostic rules that define

Table 2. Results for scene graph prediction tasks with  $n = 10$  labeled examples per predicate, reported as recall@K. A state-of-the-art scene graph model trained on labels from our method outperforms those trained with labels generated by other baselines, like transfer learning.

Baselines	Scene Graph Detection			Scene Graph Classification			Predicate Classification		
	R@20	R@50	R@100	R@20	R@50	R@100	R@20	R@50	R@100
BASELINE [ $n = 10$ ]	0.00	0.00	0.00	0.04	0.04	0.04	3.17	5.30	6.61
FREQ	9.01	11.01	11.64	11.10	11.08	10.92	20.98	20.98	20.80
FREQ+OVERLAP	10.16	10.84	10.86	9.90	9.91	9.91	20.39	20.90	22.21
TRANSFER LEARNING	11.99	14.40	16.48	17.10	17.91	18.16	39.69	41.65	42.37
DECISION TREE [38]	11.11	12.58	13.23	14.02	14.51	14.57	31.75	33.02	33.35
LABEL PROPAGATION [57]	6.48	6.74	6.83	9.67	9.91	9.97	24.28	25.17	25.41
OURS (DEEP)	2.97	3.20	3.33	10.44	10.77	10.84	23.16	23.93	24.17
OURS (SPAT.)	3.26	3.20	2.91	10.98	11.28	11.37	26.23	27.10	27.26
OURS (CATEG.)	7.57	7.92	8.04	20.83	21.44	21.57	43.49	44.93	45.50
OURS (CATEG. + SPAT. + DEEP)	7.33	7.70	7.79	17.03	17.35	17.39	38.90	39.87	40.02
OURS (CATEG. + SPAT. + WORDVEC)	8.43	9.04	9.27	20.39	20.90	21.21	45.15	46.82	47.32
OURS (MAJORITY VOTE)	16.86	18.31	18.57	18.96	19.57	19.66	44.18	45.99	46.63
OURS (CATEG. + SPAT.)	17.67	18.69	19.28	20.91	21.34	21.44	45.49	47.04	47.53
ORACLE [ $n_{ORACLE} = 108n$ ]	24.42	29.67	30.15	30.15	0.89	31.09	69.23	71.40	72.15

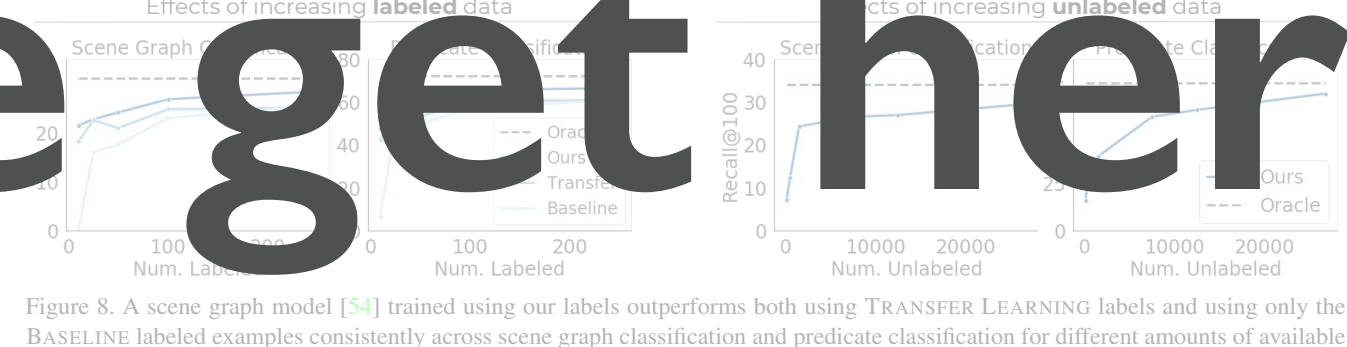


Figure 8. A scene graph model [54] trained using our labels outperforms both using TRANSFER LEARNING labels and using only the BASELINE labeled examples consistently across scene graph classification and predicate classification for different amounts of available labeled relationship instances. We also compare to ORACLE, which is trained with 108× more labeled data.

spatial features, (CATEG. + SPAT. + DEEP) combines combines all three, and OURS (CATEG. + SPAT. + WORDVEC) includes word vectors as richer representations of the cate-

objects that have a large difference in y-coordinate. In Figure 7(b), we correctly label look because phone is an important categorical feature. In some difficult cases,

# How do we get here?

into account errors in the training annotations. We adopt a noise-aware empirical risk minimizer that is often seen in logistic regression as our loss function:

$$L_\theta = \mathbb{E}_{Y \sim \pi} [\log(1 + \exp(-\theta^T V^T Y))]$$

where  $\theta$  is the learned parameters,  $\pi$  is the distribution learned by the generative model,  $Y$  is the true label, and  $V$  are features extracted by any scene graph prediction model.

We compare to a strong frequency baselines: (FREQ) uses the object counts as priors to make relationship predictions, and FREQ+OVERLAP increments such counts only if the bounding boxes of objects overlap. We include a TRANS-

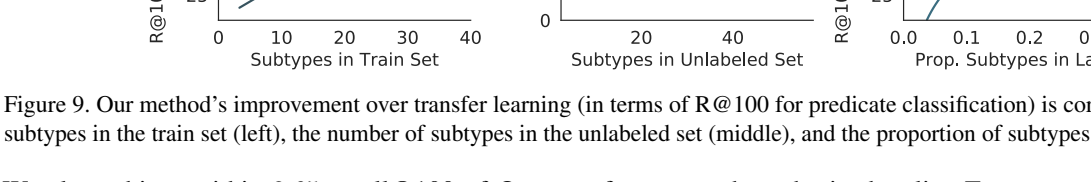


Figure 9. Our method’s improvement over transfer learning (in terms of R@100 for predicate classification) is correlated with the number of subtypes in the train set (left), the number of subtypes in the unlabeled set (middle), and the proportion of subtypes in the train set (right).

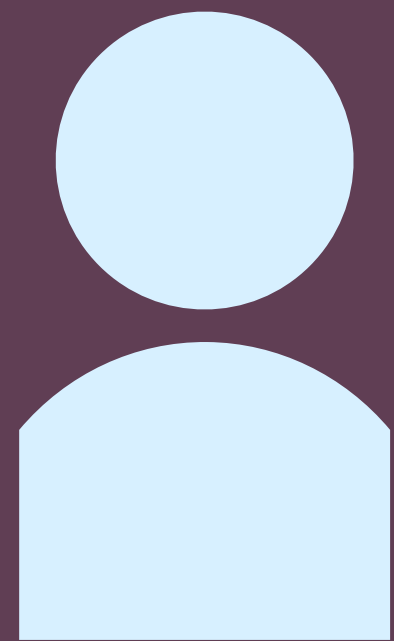
We also achieve within 8.65 recall@100 of ORACLE for SGMDET. We generate higher quality training labels than DECISION TREE and LABEL PROPAGATION, leading to an 13.83 and 22.12 recall@100 increase for PREDCLS. Effect of labeled and unlabeled data. In Figure 8 (left two graphs), we visualize how SGCLS and PREDCLS performance varies with the number of labeled examples from  $n = 2$  to  $n = 100, 50, 25, 10$ . We observe a clear advantage over TRANSFER LEARNING as  $n$  decreases with an increase of 5.16 recall@100 PREDCLS when  $n = 10$ . This result matches our observations from Section 3 because a larger set of labeled examples gives more information about a larger proportion of subtypes for each relationship. In Figure 8 (right two graphs), we visualize our performance as the number of unlabeled data points increase, finding that we approach ORACLE performance with more unlabeled examples.

Ablations. OURS (CATEG. + SPAT. + DEEP) hurts performance by up to 7.51 recall@100 for PREDCLS because it overfits to image features while OURS (CATEG. + SPAT.) performs the best. We show improvements of 0.71 recall@100 for SGMDET over OURS (MAJORITYVOTE), indicating that the generated heuristics indeed have different accuracies and should be weighted differently.

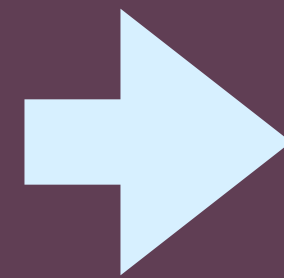
## 6. Conclusion

We introduce the first method to automatically generate knowledge bases like Visual Genome from visual relationships. We define categories as image-agnostic features and use a factor-graph based generative model that uses the probabilistic labels to unlabeled images to complete VRD datasets. We show that our method performs as well as ORACLE in F1 score when trained on a fraction of labeled data. We show that our metric to characterize the complexity of relationships is a strong indicator of how well our method performs compared to such baselines.

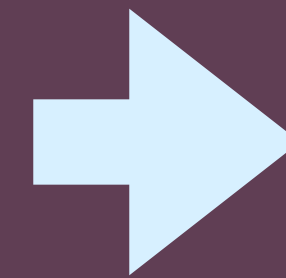
# The common malpractice



OK, time to write.



work  
work  
work  
work  
coffee  
work  
work  
work  
imposter syndrome  
work



Why is this malpractice? [I min with a partner]

Research papers are complex documents, with too many degrees of freedom to “just write”. Being strategic will save time and avoid dead ends.



# There are many genres

Even within areas, there exist many different genres of paper. Each genre is typically built around the claim you are making, and implies a structure to the sections and to the writing. For example:

## **We solve a problem:**

articulate the problem, explain what causes that problem and what others have done to deal with it, detail your approach, and prove that you make progress on the problem

## **We measure an**

**outcome:** explain that nobody has bothered understanding how a phenomenon behaves, explain how to create a study that sheds light, and report the outcomes of it

## **We introduce a**

**technique:** articulate a problem as above, but focus the narrative on the technique you've created, since it will generalize

# Genres imply structure

Common “We Solve A Problem” structure:

Introduction: overview and thesis

Related Work: situate your contribution relative to prior research

Approach: describe your approach and important implementation details

Evaluation: test whether your approach succeeds at its stated goals

Method

Results

Discussion: reflect on limitations, implications, and future work

Conclusion: summarize and restate your contribution

*But, this will vary  
by area!*



# “Which genre is our project?”

You can often derive the appropriate genre in the same way that you derived the evaluation — what is the thesis and claim that you are supporting?

But this may be challenging until you’ve read a large number of papers. So instead...

# Model papers

**A model paper is a paper that you can use as a model or template for constructing your paper.**

You should be able to structure your paper in the same way as your model paper

- Follow its general flow of argument in the introduction

- Use similar section and subsection heading organization

- Create figures, tables, and graphs that fulfill the same function as theirs

- Apply the same general proportions, e.g., number of pages per section

# Selecting your model paper

Model paper  $\neq$  nearest neighbor paper

The model paper should be a paper that makes the same type of argument as yours. It should be in the same genre as you seek.

Often the nearest neighbor paper will make a similar form of argument, but not necessarily

Often the nearest neighbor paper will be a well-written paper, but not necessarily

Find your model paper and share it with your TA for a thumbs up before writing.

# From model to paper

Start by outlining the model paper:

How does it structure its argument into sections?

What is the main expository goal of each section? What is its sub-thesis?

What role does each figure play?

# From model to paper

Next, build a mapping from their outline to yours.

Translate each section and sub-section heading into what the equivalent heading is for you

Translate each sub-thesis into what the equivalent sub-thesis is for you

Translate each figure into what the equivalent figure is for you

# What if it doesn't quite fit?

Model papers should be templates, not straightjackets. You will probably need to adapt your mapping slightly from what your model paper does.

e.g., you require a slightly different evaluation structure or visualization than them

e.g., you're drawing on a different literature than them, and need to explain something that they didn't

You can play with the genre — just don't discard the genre. Check with your TA for any substantial changes that you want to make.

# Research career paths

**“OK, so I took CS 197, now what?”**

What can you do after Stanford?

What can you do at Stanford?



# Pathways for research

Research  
is interesting



(we'll unpack this part  
in a moment)

Professor

Research scientist in industry

Entrepreneur

Engineer / Engineering Lead

# Professor

Work on research that you and the field find interesting.

Recruit the best rising talent in the world and mentor them.

Teach in your area of expertise.

Typical goals:

- Do research and have impact (e.g., publications, software adoption)

- Graduate amazing students

- Inspire students to learn about your area

- Room for personalization: entrepreneurship, speaking, consulting, &etc.

# Research scientist

Join a company's research division and work on research from within the company. Examples: Microsoft Research, FAIR, nVidia Research, Google Brain

Typical goals:

- Do research and have impact (but more focus on translation to the company's products and less on publication)

- Create innovations that transform the company you're working for (e.g., Kinect, BERT, TPUs)

# Entrepreneur

Start your own company, often based on the research you're doing, and grow it.

Typical goals:

- Scale your ideas and make them available to millions of people

- Start a new industry: your start-up is not a “me too” startup. Typically, it's pitching a dramatically new angle.

- Little focus on doing research in the short term

# Engineer / Engineering Lead

Join a company and apply your skills toward the development of product

Typical goals:

- Be the company's expert in an area, and potentially grow a team to drive product in that space

- Typically, these jobs are for types of levels of expertise and experience that cannot be acquired through a BS or MS

- Little focus on doing research in the short term

# What's the distribution?

I looked into this! I scraped names of all Ph.D. graduates in Computer Science from Stanford, MIT, and UC Berkeley.

I then mapped the names onto LinkedIn pages (yes, LinkedIn availability adds bias, but we found about 75% of people)

Tag their jobs on their LinkedIn:

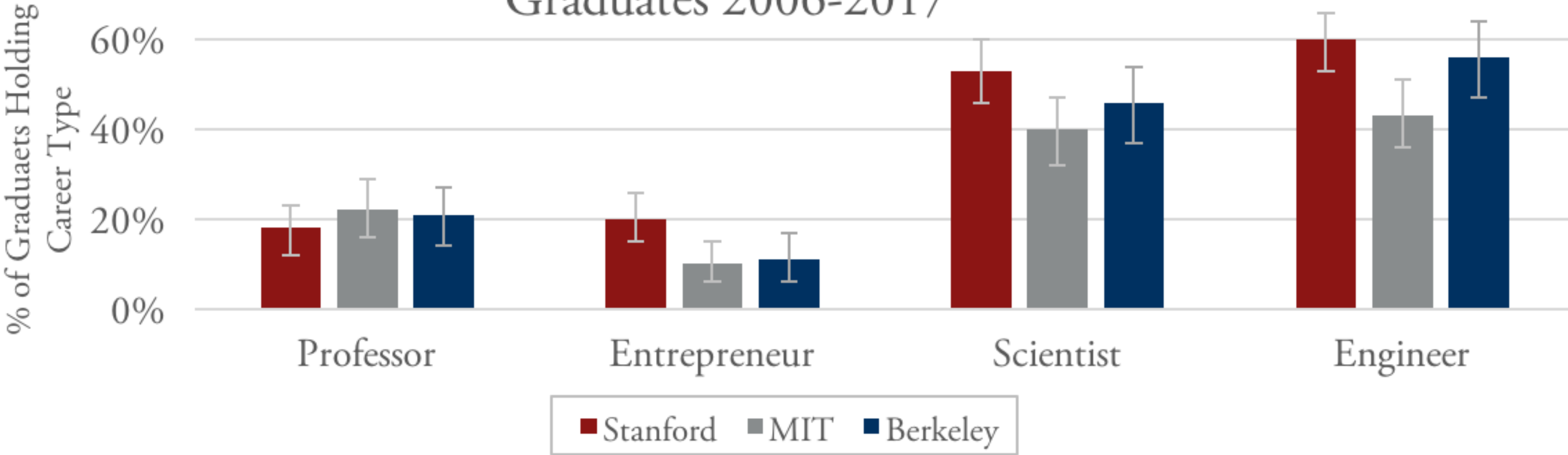
Faculty: job titles including words such as "faculty" or "professor"

Entrepreneurship: triggered by titles such as "founder" or "partner"

Research scientist: titles such as "researcher" or "scientist" (natch)

Engineer: titles such as "programmer" or "architect"

# Graduates 2006-2017



No statistically significant difference

No statistically significant difference

No statistically significant difference

Percentages add up to more than 100% because people can hold more than one position. Entrepreneurs and research scientists are a common mix. Faculty, likewise, can sometimes jump into industry research or start a company.

# Pathways for research

Research  
is interesting



(we'll unpack this part  
in a moment)

Professor

Research scientist in industry

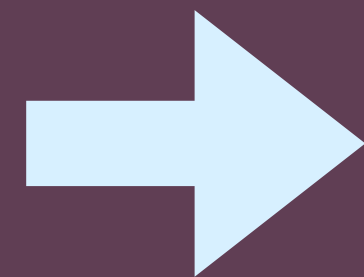
Entrepreneur

Engineer / Engineering Lead



# Pathways for research

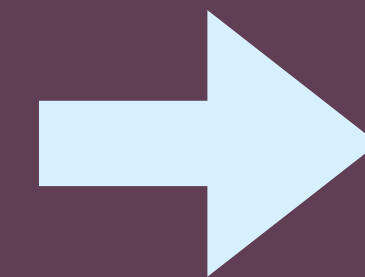
Research  
is interesting



Academic year  
research

Summer CURIS  
internship

BS with honors



Professor

Research scientist in industry

Entrepreneur

Engineer / Engineering Lead

# Academic year research

Get units for doing research with a faculty member

Generally, start with CS 195, which fulfills the CS Senior Project requirement, then go on to CS 199

How to get started? Talk to your TA about possible faculty to approach, and we can help facilitate an introduction.

Typically, you'll get involved in a project ongoing in the lab

# Summer CURIS research

Apply your full effort toward a fun research project for the summer

- Get mentored by a faculty member and PhD student

- Get paid

- No need to balance the project against classes

- Live on campus

Typically, you join a project that's ongoing in the faculty member's lab

Apply early in winter quarter at [curis.stanford.edu](https://curis.stanford.edu)

# BS with honors

Receive a special designation on your diploma (“BS with honors”)

Engage in a yearlong research project your senior year

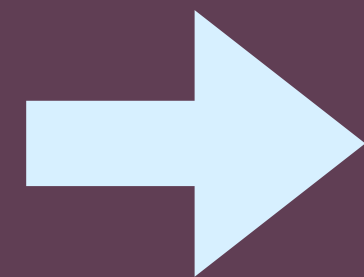
- Takes the place of the senior project

- Typically, you do this with faculty who you’ve already been working with

Apply in the spring of your junior year

# Pathways for research

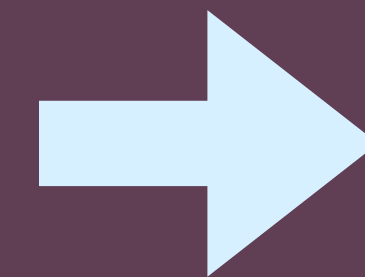
Research  
is interesting



Academic year  
research

Summer CURIS  
internship

BS with honors



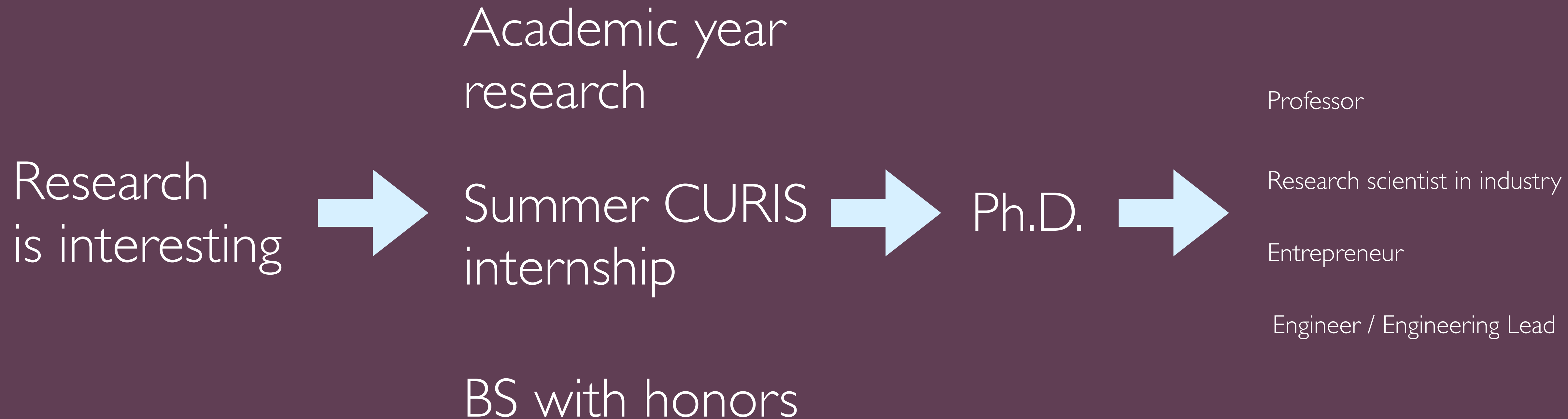
Professor

Research scientist in industry

Entrepreneur

Engineer / Engineering Lead

# Pathways for research



# All of you can succeed at a PhD!

A Ph.D. is a grown-up version of the research you do as an undergraduate or master's student. You get much more control over the projects you are working on, and become first author on the resulting publication.

It's challenging because we doubt ourselves constantly. But you also earn the ability to tackle any complex problem.

Cool side benefit: become Dr. [Lastname]

# How do I get in to a Ph.D.?

The most important criteria for getting into a Ph.D. program is **demonstrated interest and ability** to do research.

“How do I demonstrate interest and ability?” **Do research!**



# How do I get in to a Ph.D.?

In your statement, talk about research you did and the impact you had on the project. (You can include your CS 197 class project in it!)

You will want three recommendation letters from people with Ph.D.s to support your case.

Typically, one is the faculty you worked most closely with on research. The other two can be supporting letters, or other research mentors available.

**What questions  
do you have?**

# Assignment 8: draft paper

Work together with your team to write a draft paper. This should be a complete draft in the template format of your research, and include reviewable drafts of every section.

“Can we include text we already wrote?” Absolutely! + tweaks

“Do we need the results of our evaluation?” Yes, but you can continue to update your results through the final presentations.

“What if our project doesn’t work out?” Still write up the report. Negative results can be valuable. Unpack in Discussion what it was about your idea or assumptions that wasn’t borne out.

Next week, we’ll be doing mock peer review of your draft papers!

# Writing a Paper & Research Career Paths

Slide content shareable under a Creative Commons Attribution-NonCommercial 4.0 International License.