Part-Based Models for Analyzing Tooth Characters of Bat Skulls

Xu Shell Hu

EECS, Oregon State University

Abstract. This paper is a summary report for our work in the AVATOL project. We focus on tooth character scoring for bat species. Based on the needs of biologists, we propose three different scenarios of character scoring and come up with three different methods that make use of the structure among object parts (i.e., teeth). For a single species, we use the classical deformable parts model with few modifications in parameter learning. For multiple species, we propose a new model for parsing and localizing chain structured objects, where object parts may be invisible due to partial occlusions or phenotypic variations. We model the locations of parts and the dependencies between all possible pairs of parts using a graphical model. For a testing image, the goals are finding the locations of parts and the best chain structured parse graph. We show that exact inference using dynamic programming can be performed under the assumption that all the parse graphs are substrings of a chain graph. The last scenario is the case when only a small group of species are known. We want to score unknown species according to the knowledge learned from known species. We propose a zero-shot learning method inspired by recent development of attribute representations for image classification. The individual tooth detectors are treated as attribute classifiers in our case. We evaluate our methods on a dataset of bat skull images. The results show that our methods are able to score presence/absence characters of tooth automatically and achieve significant improvements over baseline methods.

1 Introduction

"Phenomic characters" represent a rich source of information for understanding biodiversity and evolution [1,2], especially for reconstructing the Tree of Life. For fossil species, these data are the only way to discover the evolutionary relationships among species. For living species, phenomic data contribute to our understanding of evolutionary relationships and provide a window into the complex interrelationships of form, environment, and genes. Phenomic characters include anatomical characteristics of organisms, such as presence or absence of shared or unique parts (e.g., horns, wings), shapes of parts (coiled versus straight horns) relationships between parts (e.g., that the eye is superior to the nose), and other features such as biochemistry and behavior. MorphoBank, a new web application and database allows researchers to collect and archive images collaboratively in online matrices [3]. MorphoBank now includes thousands of scores and annotated images used in evolutionary research. Columns in MorphoBank matrices represent characters, such as the presence/absence of a part (e.g., horns) or more complex relationships (e.g., distance between teeth). Rows in matrices are species.

Scoring each cell in a matrix, however, currently requires individual visual inspection by an expert, limiting the speed at which these data can be analyzed. A goal of our research, as a part of the collaborative AVATOL project[1], is to apply computer vision techniques to accelerate the character scoring process, which will significantly advance the reconstruction of the whole Tree of Life containing tens of millions of characters and species [4].

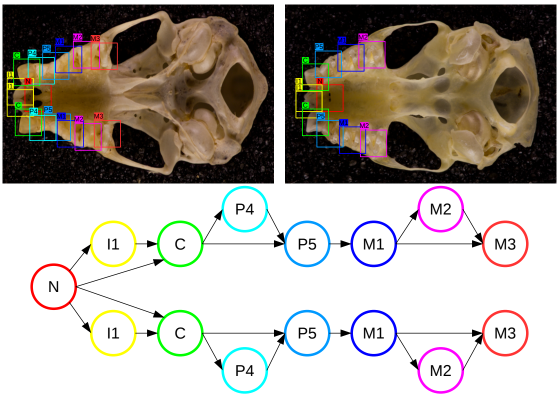


Fig. 1: Two bat skull images in ventral view and the structural dependencies between teeth. A tooth layout is a configuration of tooth types induced by a particular instance of structural dependencies.

In this work, we mainly focus on an image collection of bat skulls provided by researchers in the Department of Mammalogy at the American Museum of Natural History. Some examples of bat skulls in ventral view are illustrated in Fig. 1, Fig. 5 and Fig. 6. To analyze the evolutionary relationships between bat species, one important clue is to reason about the types and the layout of their teeth. Teeth vary widely in mammalian evolution, and their differences are important distinguishing characteristics of species and larger groups. Mammals, including bats, have four tooth types (incisors, canines, premolars, and molars). Differentiating tooth type and number can be especially problematic when teeth appear to be similar in texture and shape, in which case finding the right tooth layout could be of great help. However, the tooth layout is usually unknown unless we know in advance which species it comes from, in which case we probably know the tooth types as well. Depending on how much information we are given, we summarize the problems into three different scenarios:

1. Individual Species Scenario (ISS): We are given a set of images from a single species with ground truth annotations of tooth locations in the training set. The goal is to find the tooth layouts for all the images in the testing set;

[1] NSF – Assembling, Visualizing, and Analyzing the Tree of Life: http://avatol.org/ngp/

2. Multiple Species Scenario (MSS): We are given a database of images drawn from several bat species. All these species must appear in the training set. Given a testing image without the meta information of species, the goals are to determine tooth types and localize the teeth;
3. Zero-Shot Scenario (ZSS): The images in the testing set are drawn from species that are not included in the training set. This is the main application scenario since there are hundreds of bat species, but only a few of their images have been annotated. If all of the images were annotated, there would be no need for computer vision, because the characters would already have been scored. The goals for the test images are the same as in the MSS.

We have developed different methods for each scenario. For ISS, we model the layout of teeth as a tree graphical model with Gaussian-like pairwise potentials. The problem of localizing teeth can then be solved by exact MAP inference using dynamic programming and the distance transform. This model is known as the deformable parts model (DPM) [5] in computer vision. The proposed model for MSS is called the reconfigurable chain model (RCM), which is an extension of the DPM. Since multiple species will introduce multiple tooth layouts, we must search for the right tooth configuration while localizing the teeth at the same time. Due to the assumption of chain graphs, the inference of RCM can be solved by dynamic programming as well. The model parameters can be learned jointly using a structured SVM. For ZSS, we use a different method inspired by recent research on zero-shot learning using attribute representations [6]. In our case, we learn a set of tooth classifiers (specifically DPMs) to obtain scores under different configurations. These scores are combined to form the attribute representation, which is used to predict the tooth types.

Our experiments are performed on the existing dataset of bat skulls in Morphobank. The dataset includes 24 bat species with 10-20 images per species. For ISS, we evaluate the localization error of the teeth. For MSS and ZSS, we evaluate the prediction error of tooth types and the localization error of the teeth. Our implementations are integrated into a software package for automatic scoring of tooth characters on Morphobank with a GUI that allows users to examine the results.

The structure of this report is organized as follows. In section 2, we will talk about the technical details of deformable part models. In section 3, we present the problem under MSS and introduce the reconfigurable chain model (RCM) including its inference and parameter learning. In section 4, we discuss the zero-shot learning for character recognition and a simple method using DPMs as attribute classifiers. Our experimental results will be presented in section 5.

2 Single Species with Deformable Parts Model

For a single species, the tooth types are given, so the task is merely to localize teeth. We noticed that distinguishing among different tooth types is difficult due to the similarity in appearance. To obtain more precise localization of teeth, we make use of the prior information about the geometric layout of teeth. If we model the locations of teeth as random variables, by learning a graphical model of those locations, the localization problem can be solved by MAP inference. Similar problems have been studied in computer vision for more general objects. The most famous graphical model for object recognition due to Felzenswalb et al. [5] is called the deformable parts model (DPM). The idea of modeling object parts as a deformable pictorial structure has been studied for decades, spanning from early work by Fischler and Elschlager [7] to more recent work by Felzenszwalb et al. [5,8] and Yang et al. [9]. In the DPM, an object template is represented by a composite of part filters (i.e., the models of the appearance of each part). The geometric layout of parts is hard encoded by a tree structure, which permits exact inference. The matching of an object on the image to a pictorial structure template involves both photometric similarity measurements and structural similarity measurements.

Given a tree graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of teeth (the nodes) and $\mathcal{E}$ is the set of edges connecting neighboring teeth, the scoring function of a particular tooth layout $\mathbf{x}$ is defined as

$$f(\mathbf{x}|I) = \sum_{i \in \mathcal{V}} \theta_i(\mathbf{x}_i|I) + \sum_{(i,j)\in\mathcal{E}} \theta_{ij}(\mathbf{x}_i, \mathbf{x}_j), \tag{1}$$

where $\theta_i(\mathbf{x}_i|I)$ is the unary potential for measuring appearance likelihood of the ith tooth, and $\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ is the pairwise potential for scoring the spatial offset between tooth i and tooth j. Thus, part detection and localization amounts to estimating $\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} f(\mathbf{x}|I)$. We specify $\theta_i(\mathbf{x}_i|I)$ and $\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ as linear functions:

$$\theta_i(\mathbf{x}_i|I) = \mathbf{w}_i \cdot \boldsymbol{\psi}_i(\mathbf{x}_i, I), \tag{2}$$

$$\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{w}_{ij} \cdot \boldsymbol{\psi}_{ij}(\mathbf{x}_i, \mathbf{x}_j), \tag{3}$$

where $\boldsymbol{\psi}_i$ is a feature vector associated with i, and $\boldsymbol{\psi}_{ij}$ is a pairwise feature vector associated with (i, j). In particular, $\boldsymbol{\psi}_i(\mathbf{x}_i, I)$ represents the standard HOG descriptor extracted from a window centered at $\mathbf{x}_i$, and $\boldsymbol{\psi}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ measures the displacement between windows centered at $\mathbf{x}_i$ and $\mathbf{x}_j$. Specifically, let $\boldsymbol{\psi}_{ij}(\mathbf{x}_i, \mathbf{x}_j) = [dx, dy, dx^2, dy^2]$, where $[dx, dy]$ is the offset between $\mathbf{x}_j$ and the anchor location of j specified by $\mathbf{x}_i$. This then defines a Gaussian-like potential, which makes it possible to apply a distance transform [10] to speed up the message computations. Moreover, for all pairs ij, $\boldsymbol{\psi}_i(\mathbf{x}_i, I)$ and $\boldsymbol{\psi}_{ij}(\mathbf{x}_i, \mathbf{x}_j, I)$ are concatenated to form a joint feature map $\boldsymbol{\psi}$. Let $\mathbf{w}$ denote the corresponding parameter vector. Then the score function Eq. (1) can be rewritten as $f(I, S) = \mathbf{w} \cdot \boldsymbol{\psi}(\mathbf{x}, I)$.

Inference. To find the most probable location of teeth, we perform the standard max-product message passing algorithm, where the messages are computed via the generalized distance transform algorithm [10] since the state space of each random variable could be as large as the image. The time complexity of this inference is $O(LK)$, where L is the number of possible locations for each part and K is the number of parts.

Learning. Parameter learning is slightly different from the original DPM learning [5], where they use a latent SVM and the loss is merely related to correctly identifying the root location. In our case, we suffer a loss for errors in the location of each tooth. The learning is thus formulated as a structured SVM [11]. Note that it is also possible to apply maximum likelihood learning if marginal inference is used instead of MAP inference. Formally, given a dataset $\{I^n, \mathbf{x}^n\}_{n=1}^N$, consider the empirical risk minimization as the form

$$\min_{\mathbf{w}} \sum_n \varDelta\big(\mathbf{x}^n, \arg\max_{\mathbf{x}} f_{\mathbf{w}}(\mathbf{x}|I^n)\big), \tag{4}$$

where $\varDelta$ is the loss function. However, the above objective is not convex in general. We will optimize a convex upper bound and include a complexity regularization term.

$$\min_{\mathbf{w}} \frac{\lambda}{2}\|\mathbf{w}\|_2^2 + \sum_n \max_{\mathbf{x}} \big(f_{\mathbf{w}}(\mathbf{x}, |I^n) + \varDelta(\mathbf{x}^n, \mathbf{x})\big) - f_{\mathbf{w}}(\mathbf{x}^n|I^n) \tag{5}$$

There are two key components in the above optimization: 1) the user-defined loss function $\varDelta$ and 2) the loss augmented inference, i.e., $\max_{\mathbf{x}} \big(f_{\mathbf{w}}(\mathbf{x}, |I^n) + \varDelta(\mathbf{x}^n, \mathbf{x})\big)$. We specify the loss function as

$$\varDelta(\mathbf{x}, \hat{\mathbf{x}}) = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \kappa(\mathbf{x}_i, \hat{\mathbf{x}}_i), \tag{6}$$

where κ is a kernel for measuring the distance between the predicted location and the ground truth location, which can be one minus intersection-over-union (1-IOU) if we define appropriate bounding boxes to cover the entire teeth. The MAP inference algorithm mentioned above can be reused for the loss augmented inference. The only difference is that we will add $\frac{1}{|\mathcal{V}|}\kappa(\mathbf{x}_i, \mathbf{x}_i^n)$ when computing the unary potential $\theta_i(\mathbf{x}_i|I^n)$.

As in [12], we solve Eq. (5) using stochastic subgradient descent. Empirically, we set $\lambda = 1$ and run the training for 200 epochs, which ensures the convergence for all species.

3 The Multiple Species Scenario (MSS)

If there exist multiple species, we would like to learn a model that could correctly score all of them. By contrast, a DPM can work only for a single species. The MSS is a more general setting, where the dataset contains images from multiple species. However the name of the species in each testing image is not given. Note that the key assumption of MSS is that all the species will be seen in the training set. Thus we have enough information to learn a model to identify the species (thus tooth types) and predict tooth locations at the same time. In this section, we first discuss a simple baseline for MSS using multiple DPMs. Then, we propose a new model called the reconfigurable chain model (RCM) to handle the MSS and even more general structural variation issues in object recognition.

3.1 Joint Training of Multiple DPMs

Given multiple species, one could train a DPM for each species. However, the scores of different DPMs are not comparable in general, due to the different numbers of parts as well as mutually independent training. A simple solution inspired by the mixture components introduced in [5] is to train all DPMs jointly. By introducing additional biases, the scores can be adjusted, such that the scores of the truth species are larger than others.

In practice, we first learn individual DPMs. Their parameters associated with potential functions are not updated during the joint training. For each species, a bias is then added. We learn these biases jointly, which turns out to be equivalent to a simple multi-class classification with DPM scores as the input features.

3.2 Reconfigurable Chain Models for MSS

Consider the problem of localizing object parts on the image. In particular, we are interested in the case where some parts may be invisible due to occlusions or missing due to phenotypic variations. In general, this problem is intractable if the number of parts is large. The number of part configurations grows exponentially as the number of parts increases. In this work, we narrow our focus to a special case. Suppose the full configuration has K parts and forms a chain structure $(1, \dots, K)$. We require that all possible configurations of parts obey the substring assumption, that is, a configuration of parts has to be a substring of the full configuration $(1, \dots, K)$. For example, $(1, 3, K)$ is one of the substrings of $(1, \dots, K)$.

We will show that for this case there exists exact inference by dynamic programming, since the substring assumption defines an underlying order for parts, which allows us to consider only a small subset of the possible configurations of parts. Although this is a special case, it is important for many part localization problems whenever parts have distinct semantic meanings. For example, it is natural to define an order for shoulder, elbow and hand in the context of human pose estimation.

To describe the problem formally, we define a random variable $\mathbf{x}_i$ for each part to denote its location in the image. We then associate a binary discrete random variable d_{ij} to indicate whether there is a dependency between the part i and j. Now a configuration of parts can be read from the dependency graph $D = [d_{ij}]$, by collecting the endpoints of those edges with $d_{ij} = 1$ in a sequence. Without loss of generality, we assume the dependency graph is a complete graph. One can also eliminate impossible dependencies to form a sparser graph. An example of the dependency graph is shown in Fig.2.

Note that our problem is similar to the problem of dependency parsing in NLP, except that we are looking for a chain subgraph from the dependency graph instead of a tree. However, the dependency parsing problem does not ask for finding the state (i.e., the part location in our case) of each node in the graph. In addition, the dependencies between parts may vary for different part locations. Thus, we have to optimize over $\mathbf{x}$ and D jointly.

In case all parts are invisible, we add a virtual head node indexed by 0, but $\mathbf{x}_0$ does not represent any actual location on the image. In order to keep consistency, all the potential functions with $\mathbf{x}_0$ as inputs return 0's as the function outputs. Now, we formulate the problem as a structured prediction problem with the scoring function defined as

$$f(\mathbf{x}, D|I) = \sum_{i=0}^{K} \sum_{j=i+1}^{K} d_{ij} \left(\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \theta_j(\mathbf{x}_j|I)\right) + \left(\sum_{t=i+1}^{K} d_{it} - d_{ij}\right) \cdot \delta_{ij} \quad (7)$$

$$= \sum_{i=0}^{K} \Bigg[\sum_{j=i+1}^{K} d_{ij}(\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \theta_j(\mathbf{x}_j|I)) - \sum_{j=i+1}^{K} d_{ij} \cdot \delta_{ij} + \left(\sum_{j=i+1}^{K} d_{ij}\right) \cdot \left(\sum_{j=i+1}^{K} \delta_{ij}\right) \Bigg]. \quad (8)$$

where θ_{ij} and θ_j are unary and pairwise potentials depending on the locations of parts given the image data I, and δ_{ij} is introduced as a compensation for the unary potential of part j as well as the pairwise potential between part i and part j. Note that if $\sum_{t=i+1}^{K} d_{it} = 0$, the contribution of part i to f will be 0, since it does not make any sense to add δ_{it}'s in this case.

In order to make sure a configuration of parts is a chain, we impose additional constraints on D:

(a) Upper triangle: If $d_{ij} = 1$, then $i < j$, and $d_{ij} = 0$, whenever $i \geq j$;
(b) Single dependence: $\sum_{j=1}^{K} d_{ij} \leq 1, \forall i$ and $\sum_{i=1}^{K} d_{ij} \leq 1, \forall j$;
(c) Substring: If $d_{ij} = 1$, we have $d_{kt} = 0, \forall t$, whenever $i < k < j$ and $\sum_{t=1}^{K} d_{jt} = 1$.

Let $\mathcal{D}_c = \{D\colon D$ satisfies (a), (b) and (c)$\}$ denote the set of valid D's. Now, we could formulate the problem of parsing and localizing parts as the following discrete optimization:

$$\max_{\mathbf{x}, D \in \mathcal{D}_c} f(\mathbf{x}, D|I). \quad (9)$$

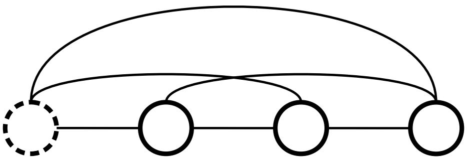


Fig. 2: The dependency graph, where the dashed node is the virtual head node.

Inference. We show that there is an exact inference procedure for maximizing over $\mathbf{x}$ and D at the same time. Although they are dependent of each other, the substring assumption allows us to decouple $\mathbf{x}$ and D in one pass from K to 0 using dynamic programming (DP).

If D is fixed, one could apply the max-product message passing algorithm to obtain the MAP solution, which is in fact a DP algorithm. On the other hand, if $\mathbf{x}$ is fixed, one could also find D exactly via another DP, which is similar to the idea of finding the maximum weighted path in a graph given the source and the sink.

We noticed that these two DPs can be operated alternately if an order of nodes is predefined, which is exactly the case when the substring assumption is made. We develop a message passing algorithm to combine them. The message passing steps are shown as follows.

$$m^1_{j\to i}(\mathbf{x}_i) = \max_{\mathbf{x}_j} \left[\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \theta_j(\mathbf{x}_j) + c_j(\mathbf{x}_j)\right], \tag{10}$$

$$m^0_{j\to i}(\mathbf{x}_i) = \max_{\mathbf{x}_j} \left[\delta_{ij} + c_j(\mathbf{x}_j)\right], \tag{11}$$

$$\mathbf{d}^*_i(\mathbf{x}_i) = \operatorname*{arg\,max}_{\mathbf{d}_i:\ \|\mathbf{d}_i\|_1 \le 1} \left[\sum_{j=i+1}^{K} m^1_{j\to i}(\mathbf{x}_i)\cdot d_{ij} + m^0_{j\to i}(\mathbf{x}_i)\cdot(1-d_{ij})\right], \tag{12}$$

$$c_i(\mathbf{x}_i) = \sum_{j=i+1}^{K} m^1_{j\to i}(\mathbf{x}_i)\cdot d^*_{ij}(\mathbf{x}_i) + \delta_{ij}\cdot(\|\mathbf{d}^*_i(\mathbf{x}_i)\|_1 - d^*_{ij}(\mathbf{x}_i)), \tag{13}$$

where $\mathbf{d}_i = \{d_{ij}\}_{j=i+1}^K$. Briefly, for each edge ij, we pass two messages representing the beliefs from below conditioned on the state of d_{ij}. The messages are combined to obtain $c_i(\mathbf{x}_i)$ by maximizing over $\mathbf{d}_i$, which now gives the best score of the substring below i. Note that Eq. (12) involves solving a linear program with respect to each location. It is not difficult to show that integer solutions are guaranteed for all such linear programs.

Once we reach the node 0, we can backtrace to obtain the MAP solution. Specifically, we need two back-pointers for Eq. (10) and Eq. (12) respectively: $\mathbf{x}^{1*}_j(\mathbf{x}_i)$ and $\mathbf{d}^*_i(\mathbf{x}_i)$. We first find $\mathbf{d}_0$ from $\mathbf{d}^*_0(\mathbf{x}_0)$. If $\|\mathbf{d}_0\| = 0$, which implies all parts are invisible, then we are done. Otherwise, there must exist a node j with $\mathbf{d}_{0j} = 1$. Now, its location can be read from $\mathbf{x}_j = \mathbf{x}^{1*}_j(\mathbf{x}_0)$. Next, we obtain $\mathbf{d}_j$ from $\mathbf{d}^*_j(\mathbf{x}_j)$. This procedure is repeated until we get the solution for node K.

Proposition 1. $c_0(\mathbf{x}_0) = \max_{\mathbf{x}, D\in\mathcal{D}_c} f(\mathbf{x}, D)$.

Proof. Let $\mathbf{x}_{i:K} = (\mathbf{x}_i, \ldots, \mathbf{x}_K)$ and $D_{i:K} = (\mathbf{d}_i, \ldots, \mathbf{d}_K)$. We can define the scoring function for a subproblem as $f(\mathbf{x}_{i:K}, D_{i:K})$. The result that $c_0(\mathbf{x}_0)$ is equal to the MAP objective value can be shown by induction. The base case holds since we have $c_K(\mathbf{x}_K) = 0$. Now, let's assume that $c_i(\mathbf{x}_i) = \max_{\mathbf{x}_{i+1:K}, D_{i:K}} f(\mathbf{x}_{i:K}, D_{i:K})$ holds for each $i \ge k$. We need to show it also holds for $c_{k-1}(\mathbf{x}_{k-1})$. In fact, with the constraint $\|\mathbf{d}_{k-1}\|_1 \le 1$, we have

$$\max_{\mathbf{x}_{k:K}, D_{k-1:K}} f(\mathbf{x}_{k-1:K}, D_{k-1:K}) = \max_{\mathbf{d}_{k-1}} \max_{\mathbf{x}_{k:K}, D_{k:K}} f(\mathbf{x}_{k-1:K}, D_{k-1:K}) \tag{14}$$

$$= \max\left\{0, \max_{j\ge k} s_j(\mathbf{x}_{k-1})\right\}, \tag{15}$$

where $s_j(\mathbf{x}_{k-1})$ corresponds the case when $d_{k-1,j} = 1$, which is given by

$$s_j(\mathbf{x}_{k-1}) = \max_{\mathbf{x}_j} \big[\max_{\mathbf{x}_{j+1:K}, D_{j:K}} f(\mathbf{x}_{j:K}, D_{j:K}) + \theta_{k-1,j}(\mathbf{x}_{k-1}, \mathbf{x}_j) + \theta_j(\mathbf{x}_j) \big]$$
$$+ \sum_{i \geq k:\ i \neq j} \delta_{k-1,i} \cdot (\|\mathbf{d}_{k-1}\|_1 - d_{k-1,i}) \tag{16}$$
$$= m^1_{j \to k-1}(\mathbf{x}_{k-1}) + \sum_{i \geq k:\ i \neq j} \delta_{k-1,i} \cdot (\|\mathbf{d}_{k-1}\|_1 - d_{k-1,i}), \tag{17}$$

since $\mathbf{d}_i = 0$ for $i \in [k, j-1]$ by the constraint (c). Thus, we have

$$c_{k-1}(\mathbf{x}_{k-1}) = \max_{\mathbf{x}_{k:K}, D_{k-1:K}} f(\mathbf{x}_{k-1:K}, D_{k-1:K}). \tag{18}$$

Comparing to Eq. (13), it is easy to show that

$$c_0(\mathbf{x}_0) = \max_{\mathbf{x}_{1:K}, D_{0:K}} f(\mathbf{x}_{0:K}, D_{0:K}) = \max_{\mathbf{x}, D} f(\mathbf{x}, D). \tag{19}$$

□

A naive algorithm that enumerates all possible locations and then solves the DP for each location takes $O(L^K K^2)$. The time complexity of our algorithm is $O(L^2 K^2)$, where L is the number of possible locations for each part. If the distance transform technique [10] is applied, the time complexity can be reduced to $O(LK^2)$.

Joint Max-Margin Learning. Suppose that the potential functions are linear functions with the following forms: $\theta_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\theta}_{ij} \cdot \boldsymbol{\phi}_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ and $\theta_i(\mathbf{x}_i|I) = \boldsymbol{\theta}_i \cdot \boldsymbol{\phi}_i(\mathbf{x}_i|I)$, where $\boldsymbol{\phi}$ is the feature function. Let the parameter vector $\mathbf{w} = [\boldsymbol{\theta}, \boldsymbol{\delta}]$. Note that $f_{\mathbf{w}}(\mathbf{x}, D)$ is a polynomial function in $\mathbf{x}$ and D, however it is also a linear function in terms of $\mathbf{w}$. To see that, we can rewrite $f_{\mathbf{w}}(\mathbf{x}, D)$ as

$$f_{\mathbf{w}}(\mathbf{x}, D) = \mathbf{w} \cdot \boldsymbol{\psi}(\mathbf{x}, D), \tag{20}$$

where $\boldsymbol{\psi} = \big(\{d_{ij} \cdot (\boldsymbol{\phi}_{ij}, \boldsymbol{\phi}_j)\}_{ij}, \{\|\mathbf{d}_i\|_1 - d_{ij}\}_{ij}\big)$. Thus, the parameters can be learned jointly using the structured SVM framework. The objective function is similar to Eq. (5), except for the loss function, which is defined as

$$\Delta(\mathbf{x}^n, D^n, \mathbf{x}, D) = \sum_{i=0}^{K} \sum_{j=i+1}^{K} \kappa(\mathbf{x}^n_j, d^n_{ij}, \mathbf{x}_j, d_{ij}), \tag{21}$$

where $\kappa(\mathbf{x}^n_j, d^n_{ij}, \mathbf{x}_j, d_{ij})$ is a kernel function given by

$$\kappa(\mathbf{x}^n_j, d^n_{ij}, \mathbf{x}_j, d_{ij}) = \begin{cases} 1, & \text{if } d^n_{ij} = d_{ij} \\ \mathbf{1}\big(\frac{\text{box}(\mathbf{x}^n_j) \cap \text{box}(\mathbf{x}_j)}{\text{box}(\mathbf{x}^n_j) \cup \text{box}(\mathbf{x}_j)} \leq 0.5\big), & \text{if } d^n_{ij} \neq d_{ij}. \end{cases} \tag{22}$$

Multiple Chains. Although RCM can only find chain instances, it can be used to localize more complex object parts whenever the object can be naturally decomposed into multiple chains. Fig. 3 shows an example of the decomposition that the pictorial structure of human body parts is decomposed into five chains. Note that the associated model with these five chains is not equivalent to a tree-structured model, since certain connections are absent, for example, the connection between the hip and the leg is not considered in the model, which is replaced by two connections between a virtual node and two real nodes. Generally, a dependency graph can be attached to another dependency graph by connecting its virtual head node to any node on that graph. MAP inference for dependency parsing will be performed independently for each dependency graph. A node can additionally receive messages from other dependency graphs attached to it, which is hard coded. Note that we have to make sure the combined graph of multiple chains is loop-free. We can randomly pick a virtual head node as the root node of the combined graph. The message passing is then similar to the max-product belief propagation on trees, which guarantees to obtain the global optimum.

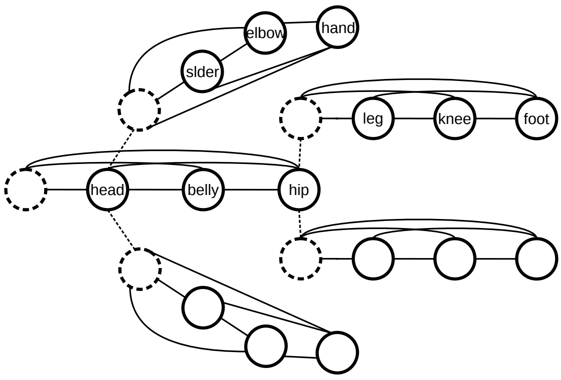


Fig. 3: An example of multiple chains for human pose estimation with structural variations.

Related Work. Recent part-based models, such as the deformable parts model (DPM) [5,9] and its hierarchical extensions [13,14], have been successfully applied to detect objects. However, these models underperform when objects are partially occluded [8]. This is because there is no effective mechanism in such models to handle missing parts. Part detectors always fire somewhere to form a full part configuration of the object, which causes false positives and false negatives.

Our approach is related to recent extensions of part-based models aimed at explicitly handling occlusion. These extensions are not appropriate for our problem either. This is because these methods typically estimate every part's visibility under the strong assumptions that the part is a) independent from the other object parts [15,16]; b) consistent with the visibility of neighboring parts [17,8]; c) consistent with the visibility of parent and children parts of the object's hierarchical structure [18]; or d) governed by a "grammar" of occlusion shape priors or occlusion patterns [19,20,21]. However, these assumptions do not hold in our setting.

4 The Zero-Shot Scenario (ZSS)

One issue with the MSS is that it works only for testing images from known species (i.e., species that appear in the training set). However, in AVATOL, the whole idea is for the scientists to annotate a small number of species, train the computer vision algorithm, and then apply it to score the characters in many more species. The scientists do not have time to manually annotate images from all of the species, and if they did, there would be no need for computer vision. We want to learn the rules for tooth configuration (or more precisely which tooth types are present) from a small number of known species and then apply these rules to many more species. Note that the prediction of a tooth type is to be performed at the species level, which cannot be properly handled by RCM due to possibly inconsistent predictions for different instances within the same species. Moreover, the RCM requires strong supervision, which is contrary to the ZSS. If we have already seen all the species in the training set, there is no point in predicting tooth configurations.

In the ZSS, we assume that the known and unknown bat species share many phonemic characters. In our case, this means that we assume they share the same types of teeth. It is reasonable to assume that teeth of the same type in different species will have similar relative position and appearance. This, in turn, should allow successful tooth detection and localization in unannotated images. It is natural to consider the transfer learning, which provides a framework to utilize prior knowledge, so that a model of a new class can be as effectively trained on a few training examples as in the case of standard training with many training data. Related work typically focuses on transfer learning for image classification. For example, boosting can be used for learning a feature representation that is shared by all image classes to address the issue of lacking training data for some classes [22]. Similarly, learning shared object parts [23], and shared training examples [24] have been demonstrated as successful transfer learning formulations for image classification. Transfer learning has also been used for attribute recognition by finding a shared classifier of object attributes [25,26]. However, in the case of ZSS, we have a much harder problem than transfer learning since there are zero training examples. Zero-shot learning has been introduced for image classification by mapping raw features of the unknown class to a common feature space of the known classes [6]. However, these approaches have not been extended to address object localization.

Let $\mathcal{K}$ and $\mathcal{U}$ denote the index sets for known species and unknown species respectively. A known species $k \in \mathcal{K}$ is represented as a tuple $(\{I^{n_k}\}_{n_k=1}^{N_k}, \{\mathbf{x}^{n_k}\}_{n_k=1}^{N_k}, \mathbf{y}^k)$, where $\mathbf{y}^k$ is a vector of binary variables denoting the tooth configuration. Similarly, an unknown species $u \in \mathcal{U}$ is given as $(\{I^{n_u}\}_{n_u=1}^{N_u}, \mathbf{y}^u)$. In the following, we will use the superscript k or u to indicate whether a species belongs the known set or the unknown set. We notice that the mapping $I^{n_k} \mapsto \mathbf{y}^k$ in a known species is usually different from the mapping $I^{n_u} \mapsto \mathbf{y}^u$ in an unknown species, since images in unknown species could be largely dissimilar to images in known species. Moreover, the configuration and the locations of teeth in I^{n_u} are unknown, which introduces extra challenges in making use of the information given by $\{\mathbf{x}^{n_k}\}_{n_k=1}^{N_k}$. Hence, to predict the tooth configuration for an unknown species, we need to build a common intermediate representation $\mathbf{a}$, which could be interpreted as an attribute representation [6], such that the mapping $\mathbf{a} \mapsto \mathbf{y}$

is shared among all species. Then, we can build the mapping from I to $\mathbf{y}$ by indirect mappings: $I \mapsto \mathbf{a}$ and $\mathbf{a} \mapsto \mathbf{y}$. To simplify the problem, we assume that the presence or absence of tooth types are independent of each other. Thus, the probability of the tooth type i is present given the image I is computed by

$$p(\mathbf{y}_i|I) = \sum_{\mathbf{a}_i} p(\mathbf{y}_i|\mathbf{a}_i)p(\mathbf{a}_i|I). \tag{23}$$

One could also consider the co-occurrence between tooth types by defining $p(\mathbf{y}|I)$ in a multi-label MRF framework [27].

We found that tooth detectors provide useful information for predicting the presence/absence of a tooth type. For a particular tooth type i, let θ_i^k denote the tooth detector learned by species $k \in \mathcal{K}$ that has tooth type i. This is equivalent to the unary potential of tooth type i if a DPM is pre-learned for species k. Then, the attribute representation of tooth type i of an unknown species $u \in \mathcal{U}$ given I^{n_u} can be defined as $\mathbf{a}_i^u|I^{n_u} = \{\mathbf{x}_i^k|I^{n_u}\}_{k\in\mathcal{K}:\ \mathbf{y}_i^k=1}$, where $\mathbf{x}_i^k|I^{n_u}$ is a detected instance of θ_i^k on I^{n_u}. One could also include interactions among $\mathbf{x}_i^k|I^{n_u}$'s, such as $\kappa(\mathbf{x}_i^k|I^{n_u}, \mathbf{x}_j^k|I^{n_u})$ discussed in section 2. We notice that this kind of information does improve performance. However, it is not easy to offer a clear probabilistic interpretation. We leave this question to future work.

The intuition behind the attribute representation is quite straightforward. A tooth detection is usually computed by sliding the template/subwindow everywhere on the image. A high score indicates high similarity between the template and an image subwindow. For each tooth type in an unknown species, there must be at least one known species sharing the same tooth type, thus there must be one high score in the attribute presentation. It could be the case that there exists other high scores due to noise, but such high scores usually occur in a predictable pattern. Hence, the occurrence of high scores actually offers a good feature for predicting whether a tooth type is present in the unknown species.

To compute $p(\mathbf{y}_i|I)$, we define $p(\mathbf{y}_i|\mathbf{a}_i)$ as a logistic regression model and $p(\mathbf{a}_i|I)$ as a product of posterior probabilities of tooth locations computed by each of the different tooth detectors. Specifically

$$p(\mathbf{y}_i = 1|\mathbf{a}_i) = \text{sigmoid}\Big(\sum_{k\in\mathcal{K}:\ \mathbf{y}_i^k=1} \mathbf{v}_i^k \cdot \theta_i^k(\mathbf{x}_i^k|I)\Big), \tag{24}$$

$$p(\mathbf{a}_i|I) = \prod_{k\in\mathcal{K}:\ \mathbf{y}_i^k=1} p(\mathbf{x}_i^k|I), \tag{25}$$

where $\mathbf{v}$ are the parameters of the logistic regression model and

$$p(\mathbf{x}_i^k|I) = \frac{\exp[\theta_i^k(\mathbf{x}_i^k|I)]}{\sum_{\mathbf{x}_i^k} \exp[\theta_i^k(\mathbf{x}_i^k|I)]}. \tag{26}$$

In practice, $p(\mathbf{x}_i^k|I)$ is usually very peaky. So we could just take top-M detections for computing $p(\mathbf{y}_i|I)$. Thus, for an unknown species u, we have

$$p(\mathbf{y}_i^u|\{I^{n_u}\}_{n_u=1}^{N_u}) = \prod_{n_u=1}^{N_u} p(\mathbf{y}_i^u|I^{n_u}). \tag{27}$$

Once we obtained the MAP solution $\hat{\mathbf{y}_i^u}$, the corresponding best detection will be treated as the estimate of the tooth location.

It is worthy mentioning that we do not include pairwise scores in ZSS. The main reason is that our pairwise potential is based on geometric constraints. However, a pairwise potential learned from a known species is very hard to transfer to an unknown species because we have observed wide variations in the geometry (e.g., compression, elongation, narrowing, etc.).

5 Experiments

In this section, we first introduce the bat skull dataset and our software. Then, we present our experimental results on the bat skull dataset for the three scenarios.

5.1 Bat Skull Dataset

The bat skull dataset consists of 954 images of bat skulls from 318 specimens. Each specimen is placed on black sand and imaged from three views: ventral, lateral and dorsal. Although these images have relatively uniform background, detecting certain skull parts of interest (in our case the teeth) is challenging due to low contrast in color and small differences in shape and texture. The specimens belong to a diverse set of 24 bat species, where each species has about 10-20 specimens. For each species, we randomly select 50% of the images for training and use the rest for testing.

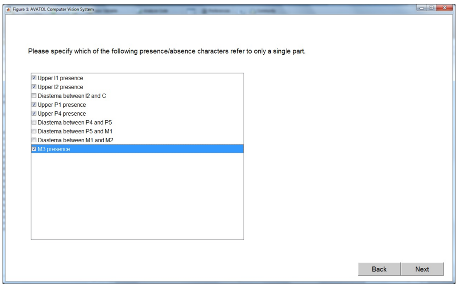
There are 9 tooth types: I1 – mesial upper incisor, I2 – distal upper incisor, C – upper canine, P1 – central upper premolar, P4 – central upper premolar, P5 – distal upper premolar, M1 – mesial upper molar, M2 – central upper molar and M3 – distal upper molar. Each species has a subset of these tooth types. Note that the C, P5, M1 and M2 tooth types are always present, so we only consider I1, I2, P1, P4 and M3 in the experiments for MSS and ZSS. Since tooth types occur in pairs, for ZSS, we only consider teeth on the right side of the jaw, in case there are confusions caused by accidental damages.

We manually annotated tooth locations for all the teeth for 318 ventral views with help from biologists. The labels of presence/absence of tooth types are provided for each species. The dataset can be downloaded from Morphobank. For access permission, please contact Andrea Cirranello (`andreacirranello@gmail.com`).

5.2 Our Software

We created a program for biologists to use who do not have machine learning background. The computer vision code is integrated into a GUI, called the AVATOL Computer Vision System, for automatic character scoring using Morphobank's matrices/dataset as input [2]. The code for the three scenarios is integrated as backends. The main program is designed to be a general character scoring tool. It requires a valid Morphobank matrix with only a few rows annotated.

From the user's perspective, the whole process can be summarized in a few steps. First, the user is asked a series of questions about image quality and other domain specific issues. The answers are collected for the data configuration and the specification of input parameters. After determining which scenario is desired, the corresponding algorithms are executed, including both training and testing. Finally, the user can use the GUI to examine the experimental results. Two screenshots are shown in Fig. 4.



Please specify which of the following presence/absence characters refer to only a single part.

- ☑ Upper I1 presence
- ☑ Upper I2 presence
- ☐ Diastema between I2 and C
- ☑ Upper P1 presence
- ☑ Upper P4 presence
- ☐ Diastema between P4 and P5
- ☐ Diastema between P5 and M1
- ☐ Diastema between M1 and M2
- ☑ M3 presence

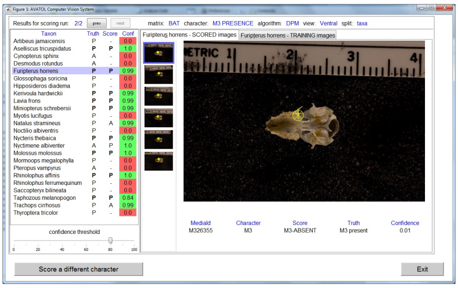


Figure 1: AVATOL Computer Vision System
prev
next
Furipterus horrens - SCORED images
Furipterus horrens - TRAINING images
confidence threshold
0
20
40
60
80
100
Score a different character
Exit

Results for scoring run: 2/2

matrix: BAT character: M3 PRESENCE algorithm: DPM view: Ventral split: taxa

Taxon	Truth	Score	Conf
Artibeus jamaicensis	P	-	0.0
Aselliscus tricuspidatus	P	P	1.0
Cynopterus sphinx	A	-	0.0
Desmodus rotundus	A	-	0.0
Furipterus horrens	P	P	0.99
Glossophaga soricina	P	-	0.0
Hipposideros diadema	P	-	0.0
Kerivoula hardwickii	P	P	0.99
Lavia frons	P	P	0.99
Miniopterus schrebersii	P	P	0.99
Myotis lucifugus	P	-	0.0
Natalus stramineus	P	A	0.99
Noctilio albiventris	P	-	0.0
Nycteris thebaica	P	P	0.99
Nyctimene albiventer	A	P	1.0
Molossus molossus	P	P	1.0
Mormoops megalophylla	P	-	0.0
Pteropus vampyrus	A	-	0.0
Rhinolophus affinis	P	P	1.0
Rhinolophus ferrumequinum	P	-	0.0
Saccopteryx bilineata	P	-	0.0
Taphozous melanopogon	P	P	0.84
Trachops cirrhosus	P	A	0.99
Thyroptera tricolor	P	-	0.0

MediaId	Character	Score	Truth	Confidence
M326355	M3	M3-ABSENT	M3 present	0.01

Fig. 4: Two screenshots of our AVATOL Computer Vision System.

5.3 Results for ISS

For ISS, we would like to test how well the DPM can detect teeth in bat skull images. We train a DPM for each species. We use the standard 32-dimensional HOG feature [28], each HOG cell has 8×8 pixels. Each part template has the same size as an image subwindow covering 5×5 HOG cells. The training includes two steps. First, part detectors are trained independently as the initialization. The parameters for pair-wise potentials are initialized to $[0, 0.1, 0, 0.1]$ as suggested by [5]. Then, part detectors are jointly trained using the structured SVM, where the loss function is the standard intersection-over-union operator proposed in the Pascal VOC challenges. We show the average loss of the detection results for each species in Table 1, which are the mean results over all tooth types. We can see that the DPM achieves pretty good results for all species. A few examples of DPM are shown in Fig. 5.

5.4 Results for the MSS

For the MSS, we propose two baselines, which are simplifications or variants of our RCM. Independent Pruning of Parts (IPP) ignores interactions between parts. Each part detector is trained independently. Detection responses are thresholded with an empirical

[2] Many thanks to Jed Irvine, who developed the GUI and the input/output system.

Species	Error	Species	Error	Species	Error
Noctilio albiventris	0.27	Aselliscus tricuspidatus	0.20	Hipposideros diadema	0.18
Artibeus jamaicensis	0.29	Cynopterus sphinx	0.25	Lavia frons	0.26
Desmodus rotundus	0.23	Kerivoula hardwickii	0.29	Miniopterus schrebersii	0.21
Glossophaga soricina	0.27	Natalus stramineus	0.18	Myotis lucifugus	0.22
Molossus molossus	0.25	Pteropus vampyrus	0.31	Nycteris thebaica	0.27
Mormoops megalophylla	0.28	Rhinolophus ferrumequinum	0.22	Nyctimene albiventer	0.22
Saccopteryx bilineata	0.24	Taphozous melanopogon	0.19	Rhinolophus affinis	0.29
Trachops cirrhosus	0.28	Furipterus horrens	0.23	Thyroptera tricolor	0.16

Table 1: The result for the ISS using DPM. The average detection error per species is measured by $1-$intersection-over-union for the bounding boxes. We report the mean over all tooth types.

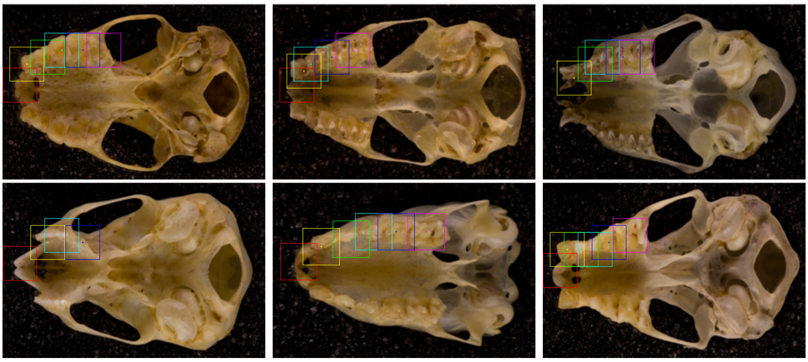


Fig. 5: A few examples of the results of the ISS. Different colors are used for different tooth types. We can see that the DPM can capture different tooth appearances and structures for different species.

threshold, estimated as the minimum score of hypotheses which highly overlap with true positives in the training set. In testing, a part will be pruned if the best score is below the empirical threshold. Multiple DPMs (MDPM) is a variant of global mixture components in DPM [5]. Each part configuration that appears in training set can be treated as a mixture component. All collected mixture components are jointly learned using SSVM. To make different components comparable, we introduce a global bias term for each mixture component. For inference, we test each image with all mixture components and take the highest scoring component as the solution of the part layout.

In the experiment, we use a fixed split for training and testing, such that training images are stratified by species. For the split, half of the images are used for training, and half of the images are used for testing. The results for all species are evaluated in terms of the detection loss and the structural loss defined in Eq. (22). The comparison of average performance for each tooth type is shown in Table 2. It can be seen that RCM outperforms IPP and MDPM in terms of both the average structural loss and the detection loss. MDPM works slightly better in predicting tooth configuration than IPP but its detection performance is affected by incorrect predictions of tooth presence.

On the other hand, IPP gains better performance in tooth detection, however, which could not help to improve the structural prediction since the dependencies between parts are ignored. Moreover, the pruning method in IPP is suboptimal, since the pruning thresholds are not learned. Fig. 6 shows some qualitative results of RCM, where missing teeth are represented with dashed bounding boxes, and red color indicates whenever a prediction is mistaken or not.

Method	I1	I2	P1	P4	M3	ASL	ADL
IPP	0.27	0.51	0.43	0.25	0.17	0.33	0.45
MDPM	0.26	0.43	0.35	0.25	0.22	0.30	0.62
RCM	0.12	0.22	0.11	0.15	0.08	0.14	0.31

Table 2: The results of MSS. The structural loss for each tooth type is given individually. The last two columns show the average structural loss (ASL) and the average detection loss (ADL) over parts.

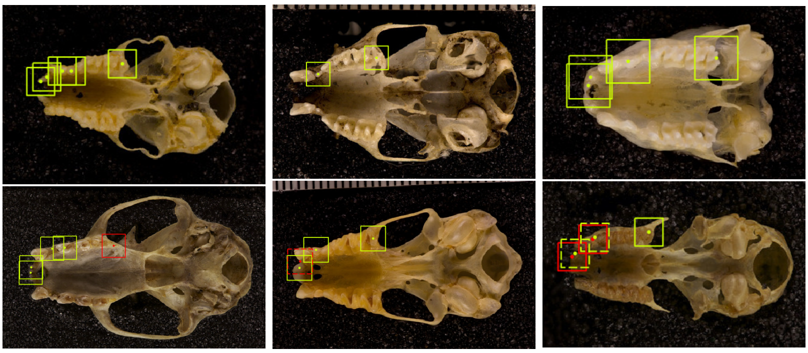


Fig. 6: The qualitative results of MSS, where bypassed teeth are represented with dashed bounding boxes, and red color indicates whenever a prediction is mistaken. We show three typical errors: non-existing teeth, bypassing teeth and incorrect tooth types.

5.5 Results for ZSS

For ZSS, we mainly focus on the prediction of tooth configurations. We do not pay particular attention to the detection results, since tooth detectors are trained in known species, and there is no supervision for adapting these detectors for unknown species. However, we find the distribution of detections is very interesting: detections are centered if a tooth type indeed exists and they are not far away from the right location. See Fig. 7 for an example. Since there are only 24 species, we manually pick 12 species as known species, and test on the remaining 12 species. The results are shown in Table 3 and Table 4. The results in Table 3 shows the performance of individual attribute classifiers $p(\mathbf{a}_i|I)$. An attribute classifier is trained for each part in each species. Then individual posterior probabilities are then combined to make the prediction for the entire species as described in section 4. Table 4 shows our experimental results of predictions of tooth type presence for the 12 unknown species. For each prediction, the number associated is the average posterior probability of tooth type presence given the images of unknown species.

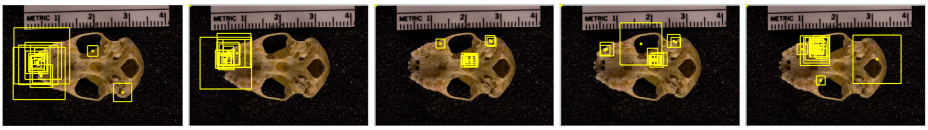


Fig. 7: The distribution of detections in unknown species by tooth detectors trained in known species.

Metrics	I1	I2	P1	P4	M3	Overall
True positive rate	0.68	0.77	0.38	0.67	0.69	0.64
True negative rate	0.69	0.62	0.70	0.60	0.70	0.66
Accuracy	0.68	0.69	0.57	0.64	0.69	0.65

Table 3: The results of ZSS with respect to individual attribute classifiers.

6 Conclusion and Future Work

In this paper, we proposed part-based models for tooth character scoring under three different scenarios. For ISS, we modeled the layout of teeth as a tree graphical model with Gaussian-like pairwise potentials. This permits the problem of localizing teeth to be solved by exact MAP inference using dynamic programming and the distance transform. For MSS, we proposed the reconfigurable chain model, which can be used as basic components whenever a problem involves solving structural variations of object parts under the substring assumption. For ZSS, we proposed a zero-shot learning method inspired by recent development of attribute representations for image classification. The evaluation is performed on the bat skull dataset downloaded from the Morphobank. The results demonstrated the effectiveness of our methods.

Recently, we have done experiments with convolutional neural networks for learning unary potentials. The accuracy is even better than the one with help from the DPM. The astonishing result suggests one of our future directions can be the combination

Species	I1	I2	P1	P4	M3	# incorr
Molossus molossus	1	0	0	0	1	1
	0	0	0	0	1	
	0.64	0.52	0.99	0.70	0.89	
Trachops cirrhosus	1	1	0	1	1	1
	1	1	1	1	1	
	0.98	1.00	0.73	0.86	0.64	
Aselliscus tricuspida-tus	1	0	1	0	1	3
	0	0	0	0	0	
	0.68	0.56	1.00	0.67	0.87	
Kerivoula hardwickii	1	1	1	1	1	1
	1	1	0	1	1	
	0.60	1.00	0.51	0.76	0.90	
Natalus stramineus	1	1	1	1	1	1
	1	1	0	1	1	
	0.70	0.64	0.56	0.78	0.53	
Taphozous melano-pogon	0	0	1	0	1	1
	0	0	0	0	1	
	0.98	0.67	0.59	0.66	1.00	

Species	I1	I2	P1	P4	M3	# incorr
Furipterus horrens	1	1	0	1	1	2
	1	1	0	0	0	
	0.68	0.65	0.82	0.64	0.83	
Lavia frons	0	0	0	0	1	1
	0	0	0	1	1	
	0.53	0.97	0.78	0.65	0.99	
Miniopterus schrebersii	1	1	0	1	1	0
	1	1	0	1	1	
	1.00	0.91	0.81	0.51	1.00	
Nycteris thebaica	1	1	0	0	1	2
	1	0	0	1	1	
	0.62	1.00	0.84	0.65	0.52	
Nyctimene albiventer	1	0	1	1	0	3
	1	1	0	0	0	
	0.80	0.58	0.60	0.51	0.67	
Rhinolophus affinis	1	0	1	0	1	2
	1	0	1	0	0	
	0.50	0.83	0.74	0.66	0.66	

Table 4: The results for the 12 unknown species. For each species, the first line stands for ground truth states. Our prediction is shown in the second line. The third line gives the average posterior probabilities. It can be seen that most of species has only one error.

of convolutional neural networks and the tooth type predictions. One possible idea is that we treat the localization and prediction problem as a regression problem, where each possible location is a state, plus an additional state for the absence of the tooth. A similar work has been done for object detection by Sermanet et al. [29]. However, the extension can be non-trivial.

Another interesting idea we have thought about is to model the tooth locations and the presences/absences in a generative process. In other words, we could associate random variables to tooth locations, tooth sizes and the gaps between teeth. The generative process starts at a tooth, for example I1. We first sample the size of I1, and then its location. Next, we sample a neighboring tooth type as well as the gap between them. We repeat this process until all tooth types have been considered or there is no more space to add a tooth. One advantage of this idea is that the zero-shot issue can be avoid. Moreover, considering the size of the problem, even a naive MCMC inference can be accepted. We will leave this interesting idea to our future work.

References

1. Lewis, P.O.: A likelihood approach to estimating phylogeny from discrete morphological character data. Syst. Biol. **50**(6) (2001) 913–925 1
2. Wiens, J.: The role of morphological data in phylogeny reconstruction. Syst Biol **53** (1999) 653–661 1
3. O'Leary, M.A., Bloch, J.I., Flynn, J.J., Gaudin, T.J., Giallombardo, A., Giannini, N.P., Goldberg, S.L., Kraatz, B.P., Luo, Z.X., Meng, J., Ni, X., Novacek, M.J., Perini, F.A., Randall,

Z., Rougier, G.W., Sargis, E.J., Silcox, M.T., Simmons, N.B., Spaulding, M., Velazco, P.M., Weksler, M., Wible, J.R., Ciranello, A.L.: Response to comment on "the placental mammal ancestor and the post-k-pg radiation of placentals". Science **341** (2013) 613 1
4. Wilson, E.O.: The encyclopedia of life. Trends in Ecology and Evolution **18** (2003) 77–80 2
5. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. PAMI (2010) 3, 4, 6, 10, 14, 15
6. Lampert, C.H., Nickisch, H., Harmeling, S.: Learning to detect unseen object classes by between-class attribute transfer. CVPR (2009) 3, 11
7. Fischler, M., Elschlager, R.: The representation and matching of pictorial structures. IEEE Transactions on Computers **C-22**(1) (1973) 67–92 4
8. Girshick, R.B., Felzenszwalb, P.F., McAllester, D.: Object detection with grammar models. NIPS (2011) 4, 10
9. Yang, Y., Ramanan, D.: Articulated pose estimation with flexible mixtures-of-parts. In: CVPR. (2011) 4, 10
10. Felzenszwalb, P., Hutenlocher, D.: Pictorial structures for object recognition. IJCV **61**(1) (2005) 55–79 4, 9
11. Tsochantaridis, I., Hofmann, T., Joachims, T., Altun, Y.: Support vector machine learning for interdependent and structured ouput spaces. ICML (2004) 4
12. Ratliff, N., Bagnell, J.A., Zinkevich, M.: Subgradient methods for structured prediction. AISTATS (2007) 5
13. Zhu, L., Chen, Y., Yuille, A.L., Freeman, W.T.: Latent hierarchical structural learning for object detection. In: CVPR. (2010) 10
14. Si, Z., Zhu, S.C.: Learning AND-OR templates for object recognition and detection. IEEE TPAMI **35**(9) (2013) 2189–2205 10
15. Wu, B., Nevatia, R.: Detection and tracking of multiple, partially occluded humans by Bayesian combination of edgelet based part detectors. IJCV **75**(2) (2007) 247–266 10
16. Wang, X., Han, T.X., Yan, S.: An HOG-LBP human detector with partial occlusion handling. In: ICCV. (2009) 10
17. Gao, T., Packer, B., Koller, D.: A segmentation-aware object detection model with occlusion handling. In: CVPR. (2011) 10
18. Duan, G., Ai, H., Lao, S.: A structural filter approach to human detection. In: ECCV. (2010) 10
19. Hsiao, E., Hebert, M.: Occlusion reasoning for object detection under arbitrary viewpoint. In: CVPR. (2012) 10
20. Pepik, B., Stark, M., Gehler, P.V., Schiele, B.: Occlusion patterns for object class detection. In: CVPR. (2013) 10
21. Li, B., Hu, W., Wu, T., Zhu, S.C.: Modeling occlusion by discriminative AND-OR structures. In: ICCV. (2013) 10
22. Torralba, A., Murphy, K.P., Freeman, W.T.: Sharing features: efficient boosting procedures for multiclass object detection. CVPR (2004) 11
23. Zhu, L., Chen, Y., Torralba, A., Freeman, W., Yuille, A.: Part and appearance sharing: Recursive compositional models for multi-view multi-object detection. CVPR (2010) 11
24. Lim, J.J., Salakhutdinov, R., Torralba, A.: Transfer learning by borrowing examples for multiclass object detection. NIPS (2011) 11
25. Farhadi, A., Endres, I., Hoiem, D.: Attribute-centric recognition for cross-category generalization. CVPR (2010) 11
26. Wang, Y., Mori, G.: A discriminative latent model of object classes and attributes. ECCV (2010) 11
27. Finley, T., Joachims, T.: Training structural svms when exact inference is intractable. In: ICML. (2008) 12

28. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR. (2005) 14
29. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networksa. In: CVPR. (2013) 18