Theory and Application of Neural and Graphical Models in Early Cancer Diagnostics

by Adityanarayanan Radhakrishnan

S.B. Mathematics, EECS. M.I.T. 2016

Submitted to the
Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

May 2017 [June 2017]

Author: Signature redacted
Department of Electrical Engineering and Computer Science
May 26, 2017

Certified by: Signature redacted
Caroline Uhler, Assistant Professor of EECS
May 26, 2017

Accepted by: Signature redacted
Christopher Terman, Chairman, Masters of Engineering Thesis Committee

Theory and Application of Neural and Graphical Models in Early Cancer Diagnostics
by Adityanarayanan Radhakrishnan
Submitted to the Department of Electrical Engineering and Computer Science

May 26, 2017
In Partial Fulfillment of the Requirements for the Degree of Master of Engineering in Electrical Engineering and Computer Science

ABSTRACT
With the recent availability of large training datasets and graphics processing units (GPUs), we address challenges in the application of graphical models and neural networks to prediction sensitive areas such as healthcare. We begin by presenting our work in the context of learning graphical models from biological data. Namely, we present a combinatorial perspective of Markov Equivalence Classes (MECs), which defines the size of solution spaces when attempting to learn a graphical model from data. Through our analysis, we show that the size of these MECs can be exponential with respect to features of the graph (such as average degree). We then switch contexts to address the challenge of developing interpretable complex models. Namely, we present a variational-inference-motivated neural network, *PatchNet*, that provides visual interpretability, and we present the application of our network to the Describable Textures Dataset (DTD), the ISIC-ISBI Melanoma Classification Challenge, and cell nucleus data.

1. Introduction to the Analysis of Markov Equivalence Classes

Bayesian networks or graphical models based on directed acyclic graphs (DAGs) are widely used to represent complex causal systems in applications ranging from computational biology to epidemiology, and sociology [23, 45, 51, 57]. A DAG entails a set of conditional independence relations through the Markov properties. Two DAGs are said to be *Markov equivalent* if they entail the same conditional independence relations. In general, observational data can only identify a DAG up to Markov equivalence. For statistical causal inference it is therefore important to enumerate and describe the set of Markov equivalence classes (MECs) and their sizes. If the MECs are large in size, then causal inference algorithms that operate in the space of MECs as compared to DAGs could significantly increase efficiency. However, gaining a full understanding of the causal relationships in a system with a large MEC requires many interventional experiments that deliberately and carefully alter one or more components of the system. The purpose of this paper is to recast this important combinatorial and enumerative question from statistics in the language of combinatorial optimization. This new perspective yields complexity results on the problem in general, as well as solutions to the problem in some special cases.

The problem of enumerating MECs has been studied from two fundamental perspectives: (1) enumerate all MECs on p nodes (as in [26]), and (2) enumerate all MECs of a given size (as in [25, 60, 66]). At the heart of these studies is a result of Verma and Pearl [64], which states that a MEC is determined by the underlying undirected graph (or *skeleton*) and the placement of immoralities, i.e. induced subgraphs of the form $X \to Z \leftarrow Y$. This characterization leads to a representation of an MEC by a graph with directed and undirected edges known as the *essential graph* [2] (or *cPDAG* [12] or *maximally oriented graph* [44]). In [26], Gillespie and Perlman use this characterization to identify all MECs on $p \leq 10$ nodes; namely, they fix a skeleton on p nodes, and then count the number of ways to compatibly place immoralities within the skeleton. The works [25, 60, 66] give inclusion-exclusion formulae for MECs of a fixed size by utilizing the combinatorial structure of the essential graph described in [2]. However, since essential graphs can be quite complicated, these formulae are only realizable for relatively constrained classes of MECs. In particular, [60] and [66] only consider MECs of size one, and [25] must fix the undirected edges of the essential graphs to be enumerated.

As exhibited by these results, the implementation of combinatorial enumeration techniques appears difficult from perspective (2). On the other hand, perspective (1) has only been considered

via computer-enumeration [26]. A common approach to difficult graphical structure enumeration problems is to specify a type of graph for which to solve the problem. This approach is used in such problems as the enumeration of independent sets, matchings, and colorings [20, 21, 40]. Given a graph, it can be useful to consider a refined set of combinatorial statistics each of which plays a role in the enumeration question. For instance, given a graph G researchers examine the total number of independent sets (or the *Fibonacci number* of G) [48, 49], the maximum size of an independent set (or *independence number* of G) [5, 42], and/or the number of independent sets of a fixed size [40]. These refined statistics work together to give a complete understanding of the problem of enumerating independent sets for G.

In the present paper, we initiate the combinatorial enumeration of MECs with respect to a fixed undirected graph and thereby recast this enumeration problem in the language of combinatorial optimization. For a graph G this amounts to enumerating all possible placements of immoralities within G [64]. Thus, we are interested in the following combinatorial statistics:

(1) $M(G)$, the total number of MECs on G,

(2) $m(G)$, the maximum number of immoralities on G,

(3) $m_k(G)$, the number of ways to place exactly k immoralities on G, and

(4) $M(G)_{\text{freq}} = (s_1(G), s_2(G), \ldots)$, where $s_i(G)$ denotes the number of MECs on G of size i.

The first three statistics fit together naturally in the polynomial presentation

$$M(G;x) := \sum_{k=0}^{m(G)} m_k(G)x^k, \qquad \text{since then} \qquad M(G;1) = M(G).$$

In general, computing any or all of these statistics for a given type of graph appears to be difficult. In this paper, we will prove the following theorem in support of this observation.

Theorem 1. *Given an undirected graph G, the problem of computing a DAG $\mathcal{G}$ with skeleton G and $m(G)$ immoralities is NP-hard.*

Here, we use the notion of NP-hardness as defined in [24, Chapter 5]. As with most NP-hard problems, restricting to special cases can make the problem tractable. In this paper, we will compute some or all of (1), (2), (3), and (4) for some special types of graphs that are important in both statistical and combinatorial settings. Moreover, these special cases can offer useful structural insights on the general problem. For example, it appears that the number and size of equivalence classes is guided by the number of cycles and high degree nodes in the skeleton. In order to test

and verify these types of observations, we develop a computer program for the enumeration of the combinatorial statistics (1), (2), (3), and (4) that expands on the original program of Gillespie and Perlman [26]. Using this program we cannot only verify the observations that high degree nodes and cycles in the skeleton play an important role, but we are also able to make the following interesting observation, indicating the profound role played by the underlying skeleton.

Theorem 2. *For $p \leq 10$, every connected graph G on p nodes has a unique frequency vector $M(G)_{freq}$.*

1.1. **A summary of our contributions.** The remainder of our analysis of MECs is organized as follows. In Section 2, we examine some first and fundamental examples including paths, cycles, and the complete bipartite graph $K_{2,p}$. We compute all the desired combinatorial statistics specified by (1), (2), (3), and (4) for these graphs. The first two examples exhibit an important connection to independent sets and vertex covers. In Section 3, we consider our enumeration question in the special setting of trees. Here, we derive results for stars, bistars, complete binary trees, and caterpillar graphs. The former two examples play an important role in bounding the number and size of MECs on tree graphs, and the latter two examples are fundamental to phlyogenetic modeling [70]. Following this, we identify bounds on the number of MECs on a given tree that exactly parallel the classically known bounds for independent sets in trees. We also identify tight bounds on the size of a MEC on a given tree using properties of the associated essential graphs. In Section 4, we prove Theorem 1 via a reduction of the minimum vertex cover problem. To do so, we prove a correspondence between minimum vertex covers of a given triangle-free graph G and minimum decompositions of G into non-overlapping stars, which we call *minimum star decompositions*. Using this correspondence, we can compute the number $m(G)$ for triangle-free graphs whose minimum star decompositions are isomorphic as forests. We apply this result to recover $m(G)$ for the complete bipartite graph $K_{p,p}$ and some special types of circulant graphs. In Section 5, we describe our computer program for the computation of the statistics (1), (2) (3), and (4). This program collects a variety of data on Markov equivalence classes and the skeleton of each class for all connected graphs on $p \leq 10$ nodes and for triangle-free graphs on $p \leq 12$ nodes. In particular, we compare class size and the number of MECs per skeleton to skeletal features including average degree, max degree, clustering coefficient, and the ratio of number of immoralities in the essential graph of the MEC to the number of induced 3-paths in the skeleton. Finally, we see that this program

validates Theorem 2, and we also use it to address the analogous result in the case of unconnected graphs. Since this work draws heavily on different concepts from two different fields, statistics and combinatorics, we provide an extensive review of the required concepts and definitions from both fields in the appendix.

2. Some First Examples

In this section, we provide some first examples for which we can compute all of the desired combinatorial statistics (1), (2) (3), and (4). The first two examples are the path and cycle on p nodes. Using some well-known results on the independent sets within these graphs, we can quickly obtain the desired numbers. The third example presented in this section is the graph $K_{2,p}$. Unlike the path and cycle, $K_{2,p}$ requires a more detailed analysis.

2.1. **Paths and cycles.** To compute the polynomial $M(G;x)$ and the vector $M(G)_{\text{freq}}$ for paths and cycles, we will use the notion of independent sets. We refer the unfamiliar reader to section A.1 for all the necessary definitions. In this section, we will use two well-studied combinatorial sequences, and their associated polynomial filtrations. Recall that the p^{th} Fibonacci number F_p is defined by the recursion

$$F_0 := 1 \quad F_1 := 1, \quad \text{and} \quad F_p := F_{p-1} + F_{p-2} \quad \text{for } p \geq 2.$$

The p^{th} *Fibonacci polynomial* is defined by

$$F_p(x) := \sum_{k=0}^{\lfloor \frac{p}{2} \rfloor} \binom{p-k}{k} x^k,$$

and it has the properties that $F_p(1) = F_p$ for all $p \geq 1$ and $F_p(x) = F_{p-1}(x) + xF_{p-2}(x)$ for all $p \geq 2$. Analogously, the p^{th} Lucas number L_p is given by the Fibonacci-like recursion

$$L_0 := 2 \quad L_1 := 1, \quad \text{and} \quad L_p := L_{p-1} + L_{p-2} \quad \text{for } p \geq 2.$$

The p^{th} *Lucas polynomial* is given by

$$L_0(x) := 2 \quad L_1(x) := 1, \quad \text{and} \quad L_p(x) := L_{p-1}(x) + xL_{p-2}(x) \quad \text{for } p \geq 2.$$

It is a well-known result that the independence polynomial of the path of length p, which we denote by I_p, is equal to the $(p+1)^{st}$ Fibonacci polynomial and the independence polynomial of

the p-cycle C_p is given by the p^{th} Lucas polynomial; that is to say,

$$I(I_p; x) = F_p(x) \quad \text{and} \quad I(C_p; x) = L_p(x).$$

With these facts in hand we prove the following theorem.

Theorem 3. *For the path I_p and the cycle C_p on p nodes we have that*

$$M(I_p; x) = F_{p-1}(x) \quad \textit{and} \quad M(C_p : x) = L_p(x) - 1.$$

In particular, the number of MECs on I_p and C_p, respectively, is

$$M(I_p) = F_{p-1} \quad \textit{and} \quad M(C_p) = L_p - 1,$$

and the maximum number of immoralities is

$$m(I_{p+2}) = m(C_p) = \left\lfloor \frac{p}{2} \right\rfloor.$$

Proof. The result follows from a simple combinatorial bijection. Since paths and cycles are the graphs with the property that the degree of any vertex is at most two, then the possible locations of immoralities are exactly the degree two nodes. That is, the unique head node j in an immorality $i \to j \leftarrow k$ must be a degree two node. In the path I_p, this corresponds to all $p-2$ non-leaf vertices, and for the cycle C_p this is all the vertices of the graph. Notice then that no two adjacent degree two nodes can simultaneously be the unique head node of an immorality, since this would require one arrow to be bidirected. Thus, a viable placement of immoralities corresponds to a choice of any subset of degree two nodes that are mutually non-adjacent, i.e. that form an independent set.

Conversely, given any independent set in I_p, a DAG can be constructed by placing the head node of an immorality at each element of the set and directing all other arrows in one direction. Similarly, this works for any nonempty independent set in C_p. (Notice that any MEC on the cycle must have at least one immorality since all DAGs have at least one sink node.) The resulting formulas are then

$$M(I_p; x) = I(I_{p-2}; x) = F_{p-1}(x) \quad \text{and} \quad M(C_p : x) = I(C_p; x) - 1 = L_p(x) - 1,$$

which completes the proof. □

It remains to compute the vectors $M(I_p)_{\text{freq}}$ and $M(C_p)_{\text{freq}}$ and the maximum number of immoralities $m(I_p)$ and $m(C_p)$. The formulae for these combinatorial statistics follow naturally from the description of the placement of immoralities given in Theorem 3.

Theorem 4. *The number $s_\ell(I_p)$ of MECs of size ℓ with skeleton I_p is the number of compositions $c_1 + \cdots + c_{k+1} = p - k$ of $p - k$ into $k + 1$ parts that satisfy*

$$\ell = \prod_{i=1}^{k+1} c_i$$

as k varies from $0, 1 \ldots, \lfloor \frac{p}{2} \rfloor$.

Proof. Let $\mathcal{G}$ be a DAG with skeleton I_p. We denote the Markov equivalence class of $\mathcal{G}$ by $[\mathcal{G}]$. By the proof of Theorem 3, we know that the immorality placements in $[\mathcal{G}]$ correspond to the nodes in an independent k-subset $\mathcal{I} \subset [p]$ on the subpath I_{p-2} of I_p induced by the non-leaf nodes of I_p. The induced graph of the complement of $\mathcal{I}$ is a forest of $k + 1$ paths. Since each member of $[\mathcal{G}]$ is a DAG with skeleton I_p that has no immoralities on these $k + 1$ paths, then each path contains a unique sink. Each independent k-subset yields a distinct forest of $k + 1$ paths on $[p] \backslash \mathcal{I}$, which corresponds to a unique partition of $p - k$ into $k + 1$ parts. The formula for $s_\ell(I_p)$ is then given by considering all such possible placements of sinks on each path in the forests over all independent sets. □

A similar argument using integer partitions allows us to compute the number of MECs of size ℓ on the p-cycle. We refer the reader to subsection A.4 for the unfamiliar terminology or notation.

Theorem 5. *The number of MECs of size ℓ in the p-cycle is*

$$s_\ell(C_p) = \sum_{k=1}^{\lfloor \frac{p}{2} \rfloor} \sum_{\substack{\mathbf{m} \in \mathbb{P}[p-2k+1, k, p-k], \\ \ell = \prod_{i=1}^{k} i^{m_i}}} \frac{p}{k} \binom{k}{m_1, \ldots, m_{p-2k+1}},$$

where $\mathbb{P}[j, k, n]$ denotes the partitions of n with k parts with largest part at most j.

Proof. Since C_p is a graph in which every node is degree 2, then each MEC of C_p containing k immoralities corresponds to an independent k-subset of $[p] := \{1, 2, \ldots, p\}$, and the subgraph of C_p given by deleting this k-subset consists of k disjoint paths. The size of this MEC is then the product of the lengths of these paths. So we need only count the number of such subgraphs for which this product equals ℓ.

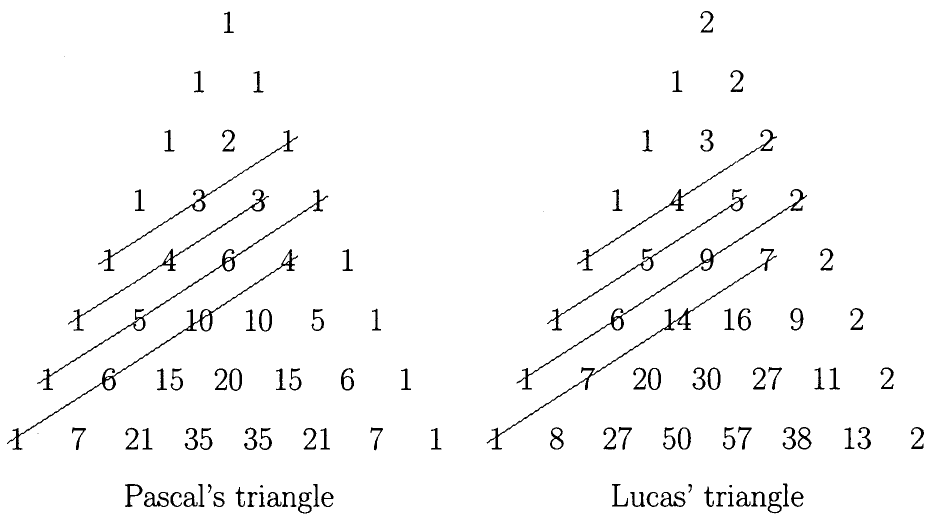


FIGURE 1. Pascal's triangle is depicted on the left and Lucas' triangle on the right. The p^{th} diagonal of each triangle is the coefficient vector of $F_p(x)$ and $L_p(x)$, respectively.

To count these objects, consider that each subgraph of C_p given by deleting an independent k-subset of C_p forms a partition of the $p-k$ remaining vertices into k parts with maximum possible part size being $p-2k+1$. Such a partition is represented by

$$\langle 1^{m_1}, 2^{m_2}, \ldots, (p-2k+1)^{m_{p-2k+1}} \rangle \in \mathbb{P}[p-2k+1, k, p-k],$$

where $m_1, \ldots, m_{p-2k+1} \geq 0$ and $\sum_i m_i = k$. The partition $\langle 1^{m_1}, 2^{m_2}, \ldots, (p-2k+1)^{m_{p-2k+1}} \rangle$ corresponds to an unlabeled forest consisting of m_i i-paths, and the number of subgraphs of C_p isomorphic to this forest is

$$\frac{p}{k}\binom{k}{m_1, \ldots, m_{p-2k+1}}.$$

The desired formula follows since the size of each corresponding MEC is $\prod_{i=1}^{k} i^{m_i}$. □

Remark 6. It is a well-known result that the coefficient of x^k in the $(p-1)^{st}$ Fibonacci polynomial is the binomial coefficient $\binom{p-k-1}{k}$, and that this is also the number of compositions of $p-k$ into $k+1$ parts. The former result says that the $(p-1)^{st}$ Fibonacci polynomial has coefficients given by the $(p-1)^{st}$ diagonal of Pascal's triangle, and so the latter result gives a compositional interpretation of the corresponding entry in Pascal's Triangle. In this subsection, we saw that this compositional interpretation of $\binom{p-k-1}{k}$ results in the proof of Theorem 4.

Analogously, the p^{th} diagonal of a second triangle, called *Lucas' triangle* in [6], corresponds to the coefficients of the p^{th} Lucas polynomial. This triangle is depicted on the right in Figure 1.

Thus, the proof of Theorem 5 results in a combinatorial interpretation of the entries of this triangle via partitions. In particular, the entry of the Lucas triangle corresponding to the k^{th} coefficient of $L_p(x)$ is

$$[x^k].L_p(x) = \sum_{\langle 1^{m_1}, 2^{m_2}, \ldots, (p-2k+1)^{m_{p-2k+1}} \rangle \in \mathbb{P}[p-2k+1, k, p-k]} \frac{p}{k} \binom{k}{m_1, \ldots, m_{p-2k+1}}.$$

Moreover, the binomial recursion on the triangle implies that these coefficients satisfy the identity

$$[x^k].L_p(x) = [x^{k-1}].L_{p-2} + [x^k].L_{p-1}.$$

To the best of the authors' knowledge, such a partition identity is new to the combinatorial literature.

2.2. **The complete bipartite graph $K_{2,p}$.** For convenience, we consider the partitioned vertex set of $K_{2,p}$ to be the two distinguished nodes $\{a, b\}$. The remaining p nodes are labeled by $[p]$ and are collectively referred to as the *spine* of $K_{2,p}$. This labeling of $K_{2,p}$ is depicted on the left in Figure 2. First, it is easy to see that the maximum number of immoralities is given by orienting the edges such that all edge heads are at the nodes a and b. This results in $m(K_{2,p}) = 2\binom{p}{2}$. Next, we compute a closed form formula for the number of MECs for $K_{2,p}$.

Theorem 7. *The number of MECs with skeleton $K_{2,p}$ is*

$$M(K_{2,p}) = \sum_{k=0}^{p} \binom{p}{k} \left(2^{p-k} - 1 + 2^k - k\right) - p2^{p-1}.$$

Proof. To arrive at the desired formula, we divide the problem into three cases:

(1) The number of immoralities at node b is $\binom{p}{2}$.

(2) The number of immoralities at node b is strictly between 0 and $\binom{p}{2}$.

(3) There are no immoralities at node b.

Notice that cases (1) and (2) have a natural interpretation via the indegree at node b of the essential graph of the corresponding MECs. Readers unfamiliar with the theory of essential graphs can find the basics in section A.3. If the indegree at b is two or more, all edges adjacent to b are essential, and the number of immoralities at node b is given by its indegree. Thus, we can rephrase cases (1) and (2) as follows:

(1) The indegree of node b in the essential graph of the MEC is p.

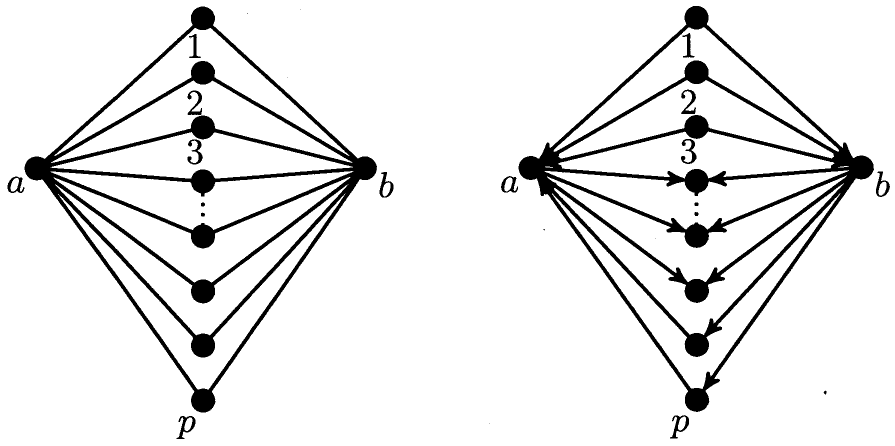


FIGURE 2. The graph $K_{2,p}$ is depicted on the left, and one of the essential graphs counted in the proof of Theorem 7 is depicted on the right.

(2) The indegree of node b in the essential graph of the MEC is $1 < k < p$.

In case (1), the MEC is determined exactly by the MEC on the star with center node a and p edges. One can easily check (this is also proven as part of Theorem 9 in the following section) that this yields $2^p - p$ MECs.

Case (2) is more subtle. First, assume that the indegree at node b is $1 < k < p$, and the arrows with head b have the tails $\{1, 2, \dots k\} \subset [p]$. Then the remaining arrows adjacent to b are all directed outwards with heads $\{k+1, \dots, p\}$. Notice that no immoralities can happen at nodes $[k]$ along the spine, but some may occur at the nodes $[p] \backslash [k]$. If there are no such immoralities, then node a has indegree p, otherwise the essential graph would contain a directed 4-cycle. Similarly, if, without loss of generality, we denote the nodes in $[p] \backslash [k]$ that are the heads of immoralities by $\{k+1, k+2, \dots, k+s\}$ for $0 \leq s < p-k$, then the nodes $k+s+1, \dots, p$ are tails of the arrows adjacent to node a. Thus, if the number of immoralities with heads in $[p] \backslash [k]$ is $0 \leq s < p-k$, then the immoralities with heads at node a are completely determined. Therefore, each s-subset of $[p] \backslash [k]$ yields a single MEC. Figure 2 depicts an example of one such choice of immoralities. We start by selecting the arrows to form immoralities at node b which forces the remaining arrows at b to point towards the spine. We then select some of these to form immoralities at the spine, and this forces the remaining arrows to be directed inwards towards a.

However, if $s = p - k$, the star induced by nodes $\{a, 1, 2, \ldots, k\}$ determines the MECs. This yields $2^k - k$ classes (see again Theorem 9). In total, for case (2) the number of MECs is

$$\sum_{k=2}^{p-1} \binom{p}{k} \left(2^{p-k} - 1 + 2^k - k\right).$$

In case (3), we consider when there are no immoralities at node b, and we count via placement of immoralities along the spine. There are 2^p ways to place immoralities along the spine, one for each subset of $[p]$. Suppose the immoralities along the spine have the heads $\{1, 2, \ldots, k\}$ for $k < p - 1$ (the cases $k = p - 1$ and $k = p$ are considered separately). Then the remaining immoralities can happen at node a. However, if there is an immorality with head at node a then all other arrows adjacent to a are essential, some of which may point towards the spine with heads in the set $[p]\backslash[k]$. Since there are no immoralities with head in the set $[p]\backslash[k]$, then any such outward pointing arrow is part of a directed path from a to b. However, since there are no immoralities at node b, there can be at most one such directed path. The presence of any such directed path forces a directed 4-cycle since $k < p - 1$. Therefore, for $k < p - 1$ the nodes $\{k + 1, \ldots, p\}$ must be tails of arrows oriented towards node a, thereby yielding only a single MEC. Since $k = p$ and $k = p - 1$ also yield only a single MEC, case (3) yields a total of 2^p classes. Combing the total number of MECs counted for each of these cases yields the desired formula. □

Using the case-by-case analysis from the proof of Theorem 7 we can count the number of MECs with skeleton $K_{2,p}$ of each possible size. Similarly, one can also recover the statistics $m_k(K_{2,p})$ from this proof. However, to avoid overwhelming the reader with formulae, we omit the expressions for $m_k(K_{2,p})$.

Corollary 8. *The possible sizes of a MEC with skeleton $K_{2,p}$ and the number of classes having each size is as follows:*

Class size	*Number of Classes*
1	$2 + \sum_{k=2}^{p-1} \binom{p}{k} 2^{p-k}$
2	$2 + \binom{p}{2}$
$3 \leq k \leq p - 1$	$1 + \binom{p}{2}$
p	2

Proof. Recall the case analysis from the proof of Theorem 7. In case (1) all MECs are size 1 except for one which is size p. This yields $2^p - p - 1$ classes of size one and one class of size p. In case (2), all MECs have size 1, unless $s = p - k$ and there are no immoralities at node a, in which case the class size is k. This yields $\binom{p}{k}$ classes of size k for $1 < k < p$, and

$$\sum_{k=2}^{p-1} \binom{p}{2} \left(2^{p-k} - 1\right)$$

classes of size 1. In case (3), all MECs have size $p - k$ for $0 \leq k < p - 1$. When $k = p - 1$, we get a single class of size 2, and when $k = p$ we get one more class of size 1. The total number of MECs of size 1 is then

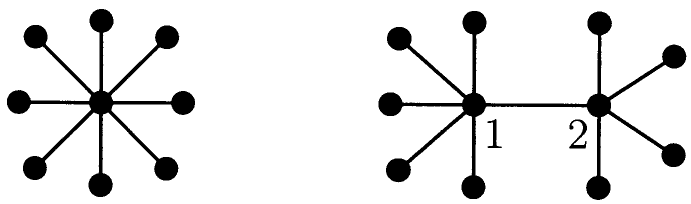
$$\begin{aligned}(2^p - p - 1) + 1 + \sum_{k=2}^{p-1} \binom{p}{k} \left(2^{p-k} - 1\right) &= (2^p - p - 1) + 1 + \sum_{k=2}^{p-1} \binom{p}{k} 2^{p-k} - \sum_{k=2}^{p-1} \binom{p}{k}, \\ &= 2^p + 2 + \sum_{k=2}^{p-1} \binom{p}{k} 2^{p-k} - \sum_{k=0}^{p} \binom{p}{k}, \\ &= 2 + \sum_{k=2}^{p-1} \binom{p}{k} 2^{p-k}.\end{aligned}$$

The other formulas are quickly realized from the above arguments. □

3. Trees

In this section, we restrict our attention to the family of trees. First we will derive formulas for some important collections of trees, including stars, that will be of significance in the coming sections. We will then provide recursions for the number of MECs for some families of trees that play important roles in phylogenetic modeling, namely caterpillar graphs and complete binary trees [70]. Following this, in Section 3.4 we will derive bounds on the number and sizes of MECs that hold for all trees. We first show that the minimum and maximum number of MECs for a tree on p nodes is achieved by the path and star graph, respectively. These results are exactly analogous to the results on independent sets in trees given in [48]. Finally, we will use the theory of essential graphs to identify tight bounds on the sizes of MECs for trees. We will see that star graphs also play an important role in achieving these bounds.

In the following, it will be helpful to label edges that have specified roles in certain Markov equivalence classes. The green edges (labeled with □) indicate that these edges cannot be involved in



$G_1(p) \simeq K_{1,p}$ $\qquad$ $G_2(p,q)$

FIGURE 3. On the left is a star and on the right is a bistar.

any immorality. The red arrows (labeled with $*$) indicate a fixed immorality in the partially directed graph, and the blue arrows (labeled with $\circ$) represent fixed arrows that are not in immoralities.

3.1. **Stars and bistars.** Two fundamental types of trees are stars and bistars. A *p-star* is simply the complete bipartite graph $K_{1,p}$, and its *center* node is the unique node of degree p. For more general purposes, we will denote the star $K_{1,p}$ as $G_1(p)$. A *bistar* can be thought of as a gluing of two stars in which a leaf node of one star is glued to the the center node of the other star. We denote the bistar given by gluing a leaf of $G_1(q+1)$ to the center node of $G_1(p)$ by $G_2(p,q)$. Equivalently, the bistar $G_2(p,q)$ can be defined by attaching p leaves to one node of the 2-path I_2 and q leaves to the other node. An example of a star and a bistar is given in Figure 3. The number of MECs on stars and their sizes will play an important role in Section 4.

Theorem 9. *The MECs on the p-star $G_1(p)$ have the polynomial generating function*

$$M(G_1(p);x) = 1 + \sum_{k\geq 2} \binom{p}{2} x^{\binom{k}{2}}.$$

In particular,

$$M(G) = 2^p - p.$$

Moreover, the corresponding class sizes are

$$s_1(G_1(p)) = 2^p - p + 1 \qquad \text{and} \qquad s_{p+1}(G_1(p)) = 1.$$

Proof. Any immorality $i \to j \leftarrow k$ in a DAG on $G_1(P)$ must have the unique head node j being the center node of $G_1(P)$, and the tail nodes i and k must be leaves of $G_1(p)$. It follows that each MEC on $G_1(p)$ having at least one immorality is given by selecting any k-subset of the p leaves for $k \geq 2$ to be directed towards the center node and then directing all other edges outwards. Each

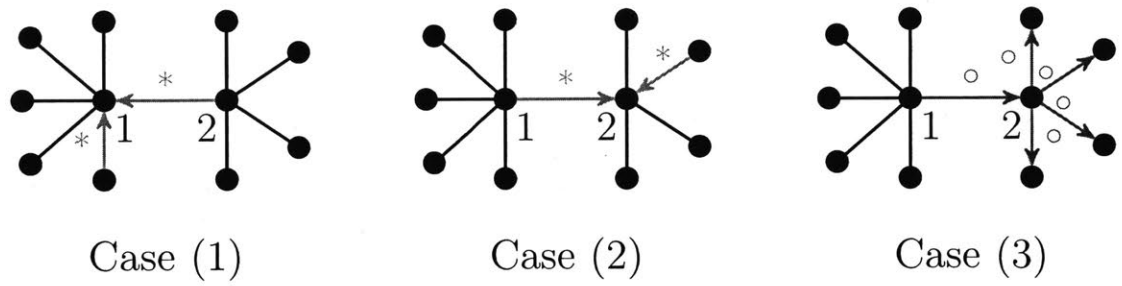


FIGURE 4. The three cases of the proof of Theorem 10.

such k-subset yields a unique MEC of size one containing $\binom{k}{2}$ immoralities. The final MEC is the class containing no immoralities. This class consists of all DAGs on $G_1(p)$ with a unique source node, and there are $p+1$ such DAGs. □

The formulas in Theorem 9 allow us to obtain similar formulas for bistars. For convenience, in the following we let

$$P_m(x) := \sum_{k=1}^{m} \binom{m}{k} x^{\binom{k+1}{2}}.$$

Theorem 10. *The MECs on the bistar $G_2(p,q)$ have the polynomial generating function*

$$M(G_2(p,q);x) = M(G_1(p);x)P_q(x) + M(G_1(q);x)P_p(x) + M(G_1(p);x) + M(G_1(q);x) - 1.$$

In particular,

$$M(G_1(p,q)) = 2^{p+q+1} - p2^q - q2^p - 1.$$

Moreover, the corresponding class sizes are

$$s_1(G_2(p,q)) = 2^{p+q+1} - p2^q - q2^p - 2^p - 2^q,$$

$$s_{p+1}(G_2(p,q)) = 2^q - 1, \qquad s_{q+1}(G_2(p,q)) = 2^p - 1, \qquad \text{and} \qquad s_{p+q+2}(G_2(p,q)) = 1.$$

Proof. To count the MECs on the bistar $G_2(p,q)$ we consider three separate cases defined in terms of the edge $\{1,2\}$. These three cases are:

(1) The edge $\{1,2\}$ is in an immorality with at least one of the p leaves attached to node 1.

(2) The edge $\{1,2\}$ is in an immorality with at least one of the q leaves attached to node 2.

(3) The edge $\{1,2\}$ is not in an immorality.

The three cases are depicted in Figure 4. In the first case, at least one of the p leaves attached

to node 1 must be in an immorality with the edge $\{1,2\}$, and the q leaves attached to node 2 can display any pattern of immoralities of the star $G_1(q)$. This yields $M(G_1(q);x)P_p(x)$ MECs as counted by their number of immoralities. Similarly, case two yields $M(G_1(p);x)P_q(x)$. In the third case, in order for the edge $\{1,2\}$ to not appear in any immorality, we need that all edges at the head of $\{1,2\}$ point towards the leaves. This yields $M(G_1(p);x) + M(G_1(q);x) - 1$ MECs as counted by their number of immoralities. Thus,

$$M(G_2(p,q);x) = M(G_1(p);x)P_q(x) + M(G_1(q);x)P_p(x) + M(G_1(p);x) + M(G_1(q);x) - 1,$$

and evaluating this polynomial at 1 yields

$$M(G_1(p,q)) = 2^{p+q+1} - p2^q - q2^p - 1.$$

Finally, to count the classes by size we again filter by the three cases (1), (2), and (3). In the first case, there are $2^p - 1$ ways for the edge $\{1,2\}$ to be in an immorality with any of the p leaves at node 1, and there are $2^q - q$ possible patterns of immoralities that can occur among the q leaves at node 2. One of these $2^q - q$ patterns has class size $q+1$ (the class with no immoralities), and all others have size one. Thus, case (1) yields $2^p - 1$ classes of size $q+1$ and $(2^q - q - 1)(2^p - 1)$ classes of size 1. Similarly, case (2) yields $2^q - 1$ classes of size $p+1$ and $(2^p - p - 1)(2^p - 1)$ classes of size 1. In case (3), if both sets of leaves contain no immoralities, then we get a single class of size $p+q+2$. If the p leaves at node 1 contain at least one immorality, then all leaves at node 2 must be directed away from node 2, yielding $2^p - p - 1$ classes of size 1. Similarly, if the q leaves at node 2 contain at least one immorality, then we get another $2^q - q - 1$ classes of size one. Summing over these cases yields the desired formulas. □

3.2. **Caterpillars.** The caterpillar graph W_p is defined to be the graph

$$W_p := \begin{cases} G_{\frac{p}{2}}(1,1,\ldots,1) & \text{if } p \text{ is even,} \\ G_{\frac{p+1}{2}}(1,1,\ldots,1,0) & \text{if } p \text{ is odd.} \end{cases}$$

The first few caterpillar graphs are depicted in Figure 5. Our definition of the caterpillar is slightly more general than the typical notion which considers only the graph $G_{\frac{p}{2}}(1,1,\ldots,1)$ for p even. The more general definition will allow us to develop a simple recursion for counting the number of MECs for this family of graphs.

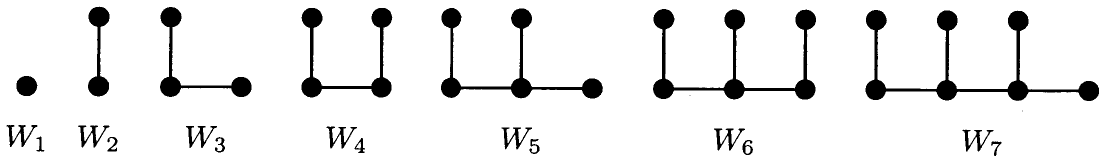


FIGURE 5. The first few caterpillar graphs.

Theorem 11. *The number of MECs for the caterpillar graph W_p is given by the recursion*

$$M(W_1) = 1, \qquad M(W_2) = 1, \qquad M(W_3) = 2, \qquad M(W_4) = 3, \qquad M(W_5) = 7,$$

and for $p \geq 6$

$$M(W_p) = \begin{cases} M(W_{p-1}) + M(W_{p-2}) & \text{if } p \text{ is even,} \\ 3M(W_{p-2}) + M(W_{p-4}) - M(W_{p-5}) & \text{if } p \text{ is odd.} \end{cases}$$

Proof. Notice first that when p is even, we can simply apply the Fibonacci recursion

$$M(G_{\frac{p}{2}}(1,1,\ldots,1)) = M(G_{\frac{p}{2}}(1,1,\ldots,0)) + M(G_{\frac{p}{2}-1}(1,1,\ldots,1)).$$

The recursion is based on whether or not the final edge is contained within an immorality.

Now let $p = 2k+1$ be odd. We first show that

$$M(W_p) = M(W_{p-1}) + 2M(W_{p-3}) + M(W_{p-2}) - \sum_{j=2}^{\lfloor \frac{p}{2} \rfloor} M(W_{p-2j-1}).$$

This recursion can be detected by considering the ways in which the final edge can or cannot be in an immorality. That is, either it is not in an immorality, or it is in an immorality with some nonempty subset of edges adjacent to it, as depicted in Figure 6. Collectively, cases (1), (2), and (3) yield

$$M(W_{p-1}) + 2M(W_{p-3})$$

MECs. On the other hand, case (4) yields $M(W_{p-2})$ minus some over-counted cases. The over-counted cases correspond to exactly when the first immorality to the right of the one depicted in case (4) points towards the right, as depicted in Figure 7. Each such case would naturally force at

least one more unspecified immorality. Thus, the total number of MECs counted by case (4) is

$$M(W_{p-2}) - \sum_{j=2}^{\lfloor \frac{p}{2} \rfloor} M(W_{p-2j-1}).$$

Since $p-1$ is even, we may apply the Fibonacci recursion to $M(W_{p-1})$ to obtain

$$M(W_p) - 2M(W_{p-2}) = 2M(W_{p-3}) - \sum_{j=2}^{\lfloor \frac{p}{2} \rfloor} M(W_{p-2j-1}).$$

We then consider the difference between $M(W_p) - 2M(W_{p-2})$ and $M(W_{p-2}) - 2M(W_{p-4})$, and repeatedly apply the Fibonacci recursion to the even terms. The result is

$$M(W_p) - 3M(W_{p-2}) + 2M(W_{p-4}) = 3M(W_{p-3}) - 4M(W_{p-5}).$$

Equivalently, we find that

$$\begin{aligned} M(W_p) - 3M(W_{p-2}) &= 3M(W_{p-3}) - 2M(W_{p-4}) - 4M(W_{p-5}), \\ &= 3M(W_{p-3}) - 2(M(W_{p-4}) + 2M(W_{p-5})), \\ &= 3M(W_{p-3}) - 2(M(W_{p-3}) + M(W_{p-5})), \\ &= M(W_{p-3}) - 2M(W_{p-5}), \\ &= M(W_{p-4}) + M(W_{p-5}) - 2M(W_{p-5}), \\ &= M(W_{p-4}) - M(W_{p-5}). \end{aligned}$$

The final equality reveals the desired recursion. □

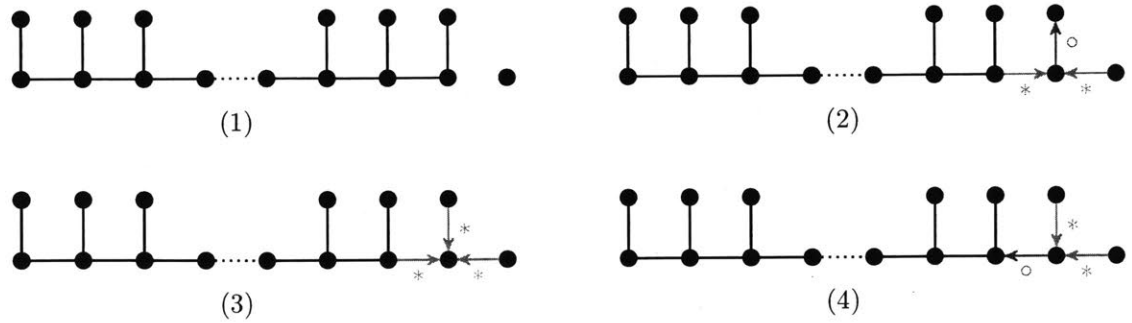


FIGURE 6. The four cases for the recursion on the caterpillar graph for p odd.

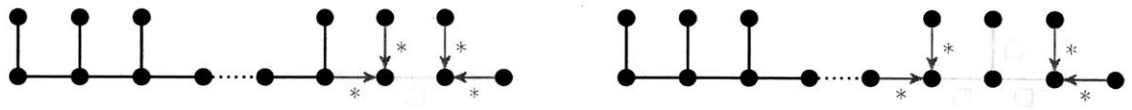


FIGURE 7. The over-counted cases of case (4) in the caterpillar recursion for p odd.

3.3. **Complete Binary Trees.** In the following, we let T_k denote the complete binary tree containing $2^k - 1$ nodes and A_k denote the additive tree constructed by adding one leaf to the root node of T_k. These two trees are depicted in Figure 8 for $k = 3$.

We will now use a series of recursions to enumerate the number of MECs on T_k and A_k. We will then show that the ratio $\frac{M(A_k)}{M(T_k)} < 4$, which means that adding an edge to the root of a complete binary tree increases the number of MECs by at most a factor of four. In practice, we observed that the factor is around two for large k.

Before providing a recursion for $M(T_k)$ and $M(A_k)$, we introduce three new graph structures X_k, Y_k, and Z_k in order to help simplify our recursions.

(1) Let X_k denote the partially directed tree whose skeleton is A_k and for which there is exactly one immorality at the child of the root (note that the root of A_k has degree 1).

(2) Let Y_k denote the number of MECs on a complete binary tree with $2^k - 1$ nodes such that the root's edges are not involved in any immoralities.

(3) Let Z_k denote the number of MECs on an additive tree with 2^k nodes such that there are edges directed from the root r to its child c and from c to each of its children.

The graphs X_3, Y_3, and Z_3 are depicted from left-to-right in Figure 9. Now we have the following series of recursions for the graphs listed above.

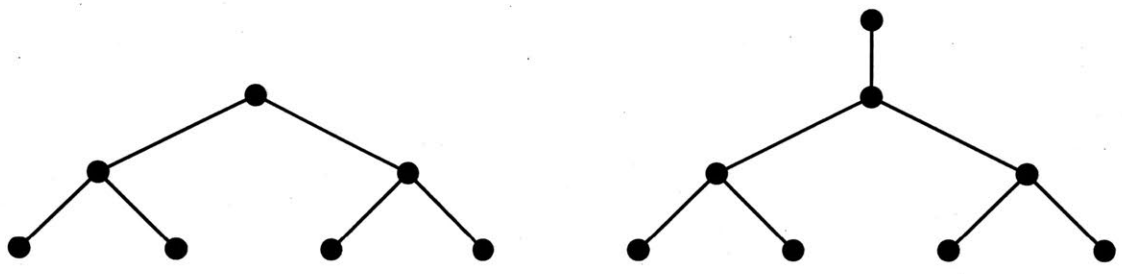


FIGURE 8. The complete binary tree T_3 is depicted on the left and the additive tree A_3 is depicted on the right.

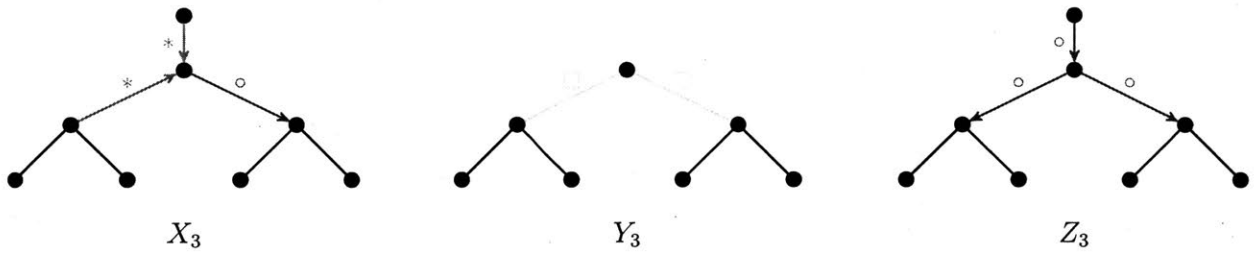


FIGURE 9. From left-to-right, the graphs X_3, Y_3, and Z_3.

Theorem 12. *The following recursions hold for the partially directed graphs T_k, A_k, X_k, Y_k, and Z_k:*

(1) $M(T_k) = M(A_{k-1})^2 + M(Y_k)$ *with* $M(T_1) = 1$,

(2) $M(A_k) = M(T_k) + 2M(X_k) + M(T_{k-1})^2$ *with* $M(A_1) = 1$,

(3) $M(X_k) = M(T_{k-1})\sqrt{M(Z_k)}$ *with* $M(X_1) = 1$,

(4) $M(Y_k) = 2M(Z_{k-1})M(T_{k-1}) - M(Z_{k-1})^2$ *with* $Y_1 = 1$, *and*

(5) $M(Z_k) = (2M(X_{k-1}) + M(T_{k-2})^2 + M(Z_{k-1}))^2$ *with* $Z_1 = Z_2 = 1$.

We first prove statements (5), (3), (4) in this order and then use them to prove statements (2) and (1).

Proof of statement (5). We prove this by analyzing the cases on the left subgraph of Z_k and consider possible immoralities at node s in Figure 10.

(1) If node s has exactly one immorality (as in the leftmost figure), then this substructure contributes exactly $M(X_{k-1})$ MECs. By symmetry, there are two ways in which node s can have exactly one immorality, which means these cases contribute $2M(X_{k-1})$ MECs.

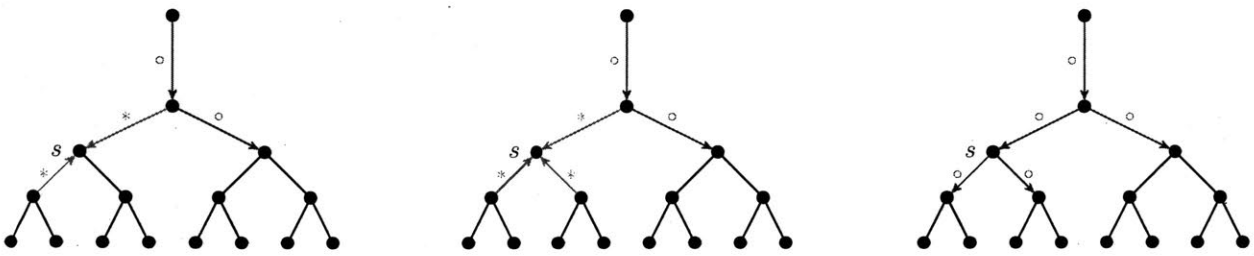


FIGURE 10.

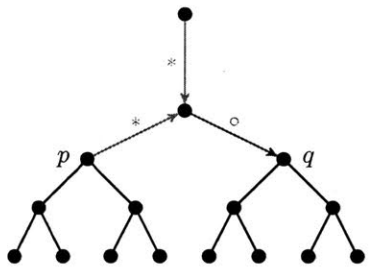


FIGURE 11.

(2) If node s has three immoralities (as in the center figure), then this substructure contributes exactly $M(T_{k-2})^2$ MECs as we may treat nodes u, v as roots of complete binary trees T_{k-2}.

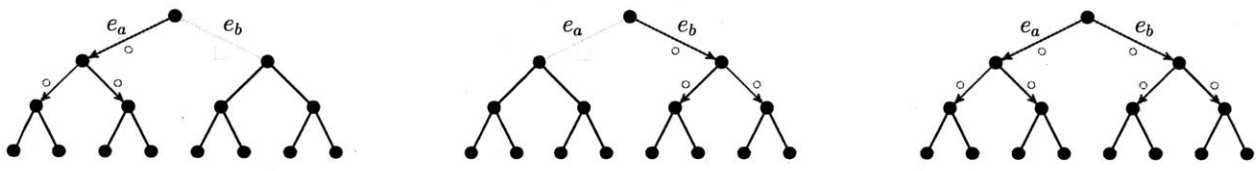
(3) If node s has no immoralities (as in the rightmost figure), then this substructure contributes exactly $M(Z_{k-1})$ MECs as we may treat the left subgraph as the graph Z_{k-1}.

Finally, as we have just considered the cases on the left subgraph of Z_k and as the immoralities on the right subgraph of Z_k are independent of the immoralities on the left subgraph, we square the number of MECs on the left subgraph to conclude $M(Z_k) = (2M(X_{k-1}) + M(T_{k-2})^2 + M(Z_{k-1}))^2$.

Proof of statement (3). Suppose we label two nodes p and q in X_k as in Figure 11. By treating node p as the root of the complete binary tree, and by treating node q as node s in the proof of statement (5), we directly have that $M(X_k) = M(T_{k-1})\sqrt{M(Z_k)}$.

Proof of statement (4). We will prove the desired recursion by considering the equivalence classes for which the edges e_a and e_b in Figure 12 are directed towards the root or away from the root.

(1) Suppose that edge e_a is directed away from the root, then edge e_b can always be directed so that it is not in an immorality at the root's right child. Thus we can consider the root's right child to be the root of the complete binary tree T_{k-1}. Now since there cannot be an



Case 1 Case 2 Case 3

FIGURE 12.

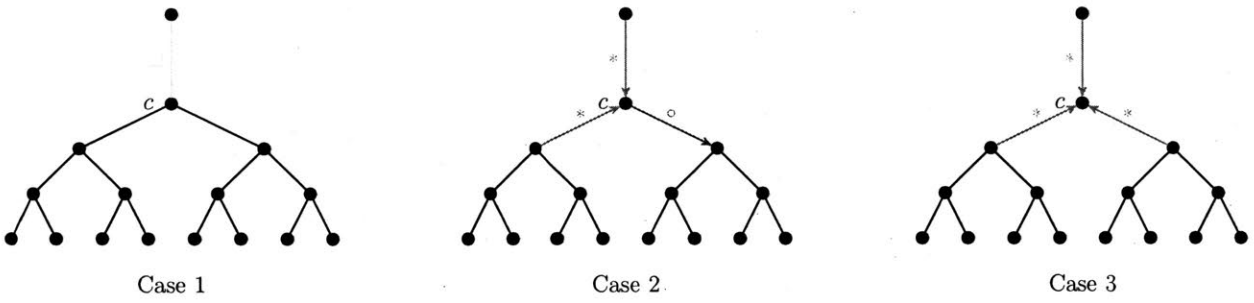


FIGURE 13.

immorality at the root's left child, the left subgraph of the root can be treated as the root of the subgraph Z_{k-1}. This case thus gives us $M(Z_{k-1})M(T_{k-1})$ MECs.

(2) Suppose that edge e_b is now directed away from the root, then this case is symmetric to the case above and so there are again $M(Z_{k-1})M(T_{k-1})$ MECs formed.

(3) In the above cases we have double-counted the cases where the edges e_a and e_b are both directed away from the root. Thus we must subtract the number of MECs formed in this case. However, in this case the left and right subgraphs from the root both represent Z_{k-1}. Thus, there are $M(Z_{k-1})^2$ MECs in this case.

Hence we have that $M(Y_k) = 2M(Z_{k-1})M(T_{k-1}) - M(Z_{k-1})^2$.

Proof of statement (2). To prove recursion (2), we will consider the three possible cases of immoralities that can occur at the child c of the root as depicted in Figure 13.

(1) In the leftmost figure, if there is no immorality formed by the edge from the root to c, then c can be treated as the root of the complete binary tree T_k. This case contributes $M(T_k)$ MECs.

(2) In the center figure, if there is exactly one immorality formed by the edge from the root to s, then the root can be treated as the root of the tree X_k. This case contributes $2M(X_k)$ MECs, as there are two ways in which the edge from the root to c can be in exactly one immorality.

(3) In the rightmost figure, if there are three immoralities formed by the edge from the root to c, then the children of c can be treated as roots of complete binary trees T_{k-1}. This case contributes $M(T_{k-1})^2$ MECs.

Thus, summing over the three cases we have that $M(A_k) = M(T_k) + 2M(X_k) + M(T_{k-1})^2$.

Proof of statement (1). We can consider the following four cases depicted in Figure 14 based on the immoralities formed by the root's edges e_a and e_b.

(1) If the edges e_a and e_b form an immorality at the root, then the root's children p and q can be treated as roots of complete binary trees T_{k-1}. This case contributes $M(T_{k-1})^2$ MECs.

(2) If the edge e_a forms at least one immorality at p but edge e_b is not in any immoralities, then edge q can be treated as the root of a complete binary tree T_{k-1}. Now p can have exactly one immorality, in which case the left subgraph of the root is the structure X_{k-1} or p can have three immoralities, in which case the children of p can each be treated as the root of a complete binary tree T_{k-2}. Now by symmetry we may consider immoralities formed by the edge e_b as well, which will double the number of MECs formed. Thus, there are $2M(T_{k-1})[2M(X_{k-1}) + M(T_{k-1})^2]$ MECs.

(3) If the edges e_a and e_b form immoralities at p and q, then by following the reasoning in the previous case, there are $2M(X_{k-1}) + M(T_{k-2})^2$ MECs formed.

(4) If the edges e_a and e_b form no immoralities, then the remaining graph is simply the structure Y_k. This case contributes $M(Y_k)$ MECs.

Summing over the different cases we have that

$$\begin{aligned} M(T_k) &= M(T_{k-1})^2 + 2M(T_{k-1})[2M(X_{k-1}) + M(T_{k-1})^2] + 2M(X_{k-1}) + M(T_{k-2})^2 + M(Y_k), \\ &= [M(T_{k-1}) + 2M(X_{k-1}) + M(T_{k-2})^2]^2 + M(Y_k), \\ &= M(A_{k-1})^2 + M(Y_k). \end{aligned}$$

This completes the proof of Theorem 12. □

Now that we have recursions for T_k and A_k, we can establish a bound on the number of MECs given by adding an edge to the root of T_k to produce A_k. In order to do this, we will use the following lemma.

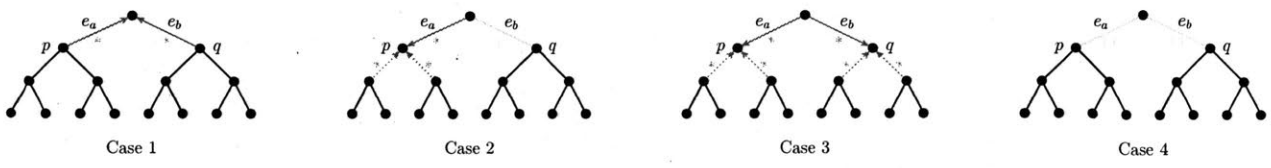


FIGURE 14.

Lemma 13. *For the partially directed graphs T_k and Z_k we have that*

$$M(Z_k) < M(T_k).$$

Proof. If we omit the root and its edge from the graph Z_k, then we see that every MEC formed in Z_k can also be formed in T_k. Further, since the MEC in T_k with an immorality at the root cannot appear in Z_k, we have a strict inequality. Hence, we have that $M(Z_k) < M(T_k)$. □

Now we show that adding an edge to the root of T_k increases the number of MECs by at most 4.

Theorem 14. *The number of MECs on A_k and T_k satisfy*

$$1 < \frac{M(A_k)}{M(T_k)} < 4.$$

Proof. First we let $R_k = \frac{M(A_k)}{M(T_k)}$ and $S_{k-1} = \frac{M(T_k)}{M(T_{k-1})^2}$. By equation 2 of Theorem 12 we know that

$$M(A_k) = M(T_k) + 2M(X_k) + M(T_{k-1})^2,$$

and hence by equation 3 of Theorem 12

$$\begin{aligned} R_k &= 1 + \frac{2M(X_k)}{M(T_k)} + \frac{M(T_{k-1})^2}{M(T_k)}, \\ &= 1 + \frac{2M(T_{k-1})\sqrt{M(Z_k)}}{M(T_k)} + \frac{M(T_{k-1})^2}{M(T_k)}. \end{aligned}$$

Thus, it follows by Lemma 13 that

$$R_k < 1 + \frac{2}{\sqrt{S_{k-1}}} + \frac{1}{S_{k-1}}$$

and hence

$$R_k < \left(1 + \frac{1}{\sqrt{S_{k-1}}}\right)^2 < \left(1 + \frac{1}{\sqrt{1}}\right)^2,$$

which completes the proof. □

3.4. **Bounding the number and size of MECs for trees.** We begin this subsection by deriving upper and lower bounds on the number of MECs for trees on p nodes. These bounds are achieved by the $(p-1)$-star $G_1(p-1)$ and the p-path I_p, respectively. This result parallels the classic result of [48], which states that the number of independent sets in a tree on p nodes is bounded by the number of independent sets in $G_1(p-1)$ and the number of independent sets in I_p, respectively.

Theorem 15. *Let T_p be a tree on p nodes. Then*

$$F_{p-1} = M(I_p) \leq M(T_p) \leq M(G_1(p-1)) = 2^{p-1} - p + 1.$$

Proof. We first prove the upper bound on $M(T_p)$. Since T_p is a tree, it has precisely $p-1$ edges, and so there are 2^{p-1} edge orientations on T_p. Of these 2^{p-1} orientations, the p orientations given by selecting a unique source node in T_p all belong to the same MEC. So there are at most $2^{p-1} - p + 1$ MECs for T_p. By Theorem 9, this bound is achieved by the $(p-1)$-star $G_1(p-1)$.

To prove the lower bound, we use a simple inductive argument. Notice first that the bound is true when $p \leq 5$. Now recall that every tree on p nodes can be constructed in one of two ways: (1) attaching a leaf to a degree 1 node of a tree on $p-1$ nodes, or (2) attaching a leaf to a node of T_{p-1} that is a neighbor of a leaf. Thus, given a tree T on $p-1$ nodes, it suffices to show that when we construct T_p from a tree T_{p-1} via (1) or (2), the number of MECs increases by at least F_{p-3}.

In case (1), we attach a leaf node v to a leaf u of T_{p-1}, whose only neighbor in T_{p-1} is some node w. The MECs on T_p then come in two types: either the edge $\{v, u\}$ is not in an immorality or it is in the immorality $v \to u \leftarrow w$. The number of classes in the first case is $M(T_{p-1})$ and the number of classes in the second case is $M(T_{p-1} \backslash u)$. So by the inductive hypothesis we have that

$$M(T_p) \geq M(T_{p-1}) + M(T_{p-1} \backslash u) \geq F_{p-2} + F_{p-3} = F_{p-1}.$$

In case (2), the leaf node v is attached to some node u of T_{p-1} that has at least one leaf w in T_{p-1}. The MECs on T_p contain two disjoint types of classes: classes in which the edge $\{v, u\}$ is not in an immorality and classes containing the immorality $v \to u \leftarrow w$. Similar to the previous case, it then follows from the inductive hypothesis that

$$M(T_p) \geq M(T_{p-1}) + M(T_{p-1} \backslash w) \geq F_{p-2} + F_{p-3} = F_{p-1},$$

which completes the proof. □

In the remainder of this section, we derive bounds on the size of the MEC for a fixed DAG $\mathcal{T}_p$ on the underlying undirected graph T_p. These bounds will be computed in terms of the structure of the essential graph $\widehat{\mathcal{T}_p}$ of the MEC $[\mathcal{T}_p]$. To see why it is reasonable to work with the essential graph to derive such bounds, recall the analysis of the MEC sizes for stars and bistars given in Theorems 9 and 10. In order to derive the possible sizes of these MECs, we implicitly counted all possible

orientations of the non-essential edges in the essential graph of each class. Since understanding the possible orientations of these edges is equivalent to knowing the size of the class, we will bound the size of the MEC of $\mathcal{T}_p$ in terms of the number and size of the chain components of $\widehat{\mathcal{T}}_p$. We will see that the computed bounds are tight, and that stars play an important role in achieving these bounds. We refer the reader to Section A.3 for the basics and notation relating to essential graphs.

In the following, we assume that the essential graph $\widehat{\mathcal{T}}_p$ has chain components $\tau_1, \tau_2, \ldots, \tau_\ell$ for $\ell > 0$. We also assume that each τ_i is *nontrivial*; i.e. it has at least two vertices. We let $\mathcal{G}(\widehat{\mathcal{T}}_p)$ denote the directed subforest of the essential graph $\widehat{\mathcal{T}}_p$ consisting of all directed edges of $\widehat{\mathcal{T}}_p$, and we let $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$ denote its connected components.

Lemma 16. *Let $\mathcal{T}_p$ be a directed tree on p nodes and $\widehat{\mathcal{T}}_p$ the corresponding essential graph. If $\widehat{\mathcal{T}}_p$ has chain components $\tau_1, \tau_2, \ldots, \tau_\ell$, then the size of the Markov equivalence class $[\mathcal{T}_p]$ is*

$$\#[\mathcal{T}_p] = \prod_{i=1}^{\ell} |V(\tau_i)|.$$

Proof. Each element of $[\mathcal{T}_p]$ corresponds to one of the ways to direct the components $\tau_1, \ldots, \tau_\ell$, each of which is a tree. Suppose we directed τ_i so that it has two source nodes s_1 and s_2. Then along the unique path between s_1 and s_2 in the directed τ_i, there must lie an immorality that is not present in $\widehat{\mathcal{T}}_p$. Thus, the only admissible directions of the components τ_i have no more than one source node. Since every DAG has at least one source node, the number of admissible directions of each τ_i is precisely the number of ways to pick the unique source node of τ_i. This is precisely the number of vertices in τ_i, thereby completing the proof. □

Lemma 16 allows us to compute the following bounds on the size of a MEC for trees.

Theorem 17. *Let $\mathcal{T}_p$ be a directed tree on p nodes and $\widehat{\mathcal{T}}_p$ the corresponding essential graph. Suppose that $\widehat{\mathcal{T}}_p$ has $\ell > 0$ chain components $\tau_1, \tau_2, \ldots, \tau_\ell$ and that the directed subforest $\mathcal{G}(\widehat{\mathcal{T}}_p)$ of $\widehat{\mathcal{T}}_p$ has $m \geq 0$ connected components $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_m$. Then*

$$2^\ell \leq \#[\mathcal{T}_p] \leq \left(\frac{p-m}{\ell}\right)^\ell.$$

Proof. Notice first that the lower bound is immediate from Lemma 16 and the assumption that each τ_i is nontrivial. So it only remains to verify the proposed upper bound.

Let ℓ_i denote the number of chain components that are adjacent to ε_i for all $i \in [m]$. Since the chain components $\tau_1, \ldots, \tau_\ell$ are all disjoint, it follows that

$$1 + \ell_i \leq |V(\varepsilon_i)|$$

for all $i \in [m]$. Therefore, a lower bound on the size of the number of nodes in the directed subforest $\mathcal{G}(\widehat{\mathcal{T}_p})$ is given by

$$m + \sum_{i=1}^{m} \ell_i \leq |V(\mathcal{G}(\widehat{\mathcal{T}_p}))|.$$

A closed form for the sum $\sum_{i=1}^{m} \ell_i$ is recovered as follows. Consider a complete bipartite graph $K_{\ell,m}$ whose vertices are partitioned into two blocks A and B where $|A| = \ell$ and $|B| = m$. The possible ways to assemble the components $\tau_1, \ldots, \tau_\ell$ and $\varepsilon_1, \ldots, \varepsilon_m$ into an essential tree are in bijection with the spanning trees of $K_{\ell,m}$. For any such spanning tree T of $K_{\ell,m}$, each edge of T has exactly one vertex in each of A and B. Thus,

$$\sum_{i=1}^{m} \ell_i = \sum_{v \in A} \deg_T(v) = \sum_{v \in B} \deg_T(v).$$

Since T is a tree, it follows that

$$\sum_{i=1}^{m} \ell_i = \frac{\sum_{v \in A} \deg_T(v) + \sum_{v \in B} \deg_T(v)}{2} = \ell + m - 1. \tag{3.1}$$

Therefore,

$$2m + \ell - 1 \leq |V(\mathcal{G}(\widehat{\mathcal{T}_p}))|.$$

Moreover, since $\mathcal{T}_p$ has p vertices, and each edge of a spanning tree of $K_{\ell,m}$ corresponds to exactly one of the vertices shared by $\mathcal{G}(\widehat{\mathcal{T}_p})$ and the chain components $\tau_1, \ldots, \tau_\ell$, then we have that

$$\sum_{j=1}^{\ell} |V(\tau_j)| = p + m + \ell - 1 - |V(\mathcal{G}(\widehat{\mathcal{T}_p}))|. \tag{3.2}$$

Now by Lemma 16 and the arithmetic-geometric mean inequality, we have

$$\#[\mathcal{T}_p] = \prod_{j=1}^{\ell} |V(\tau_j)| \leq \left(\frac{\sum_{j=1}^{\ell} |V(\tau_j)|}{\ell} \right)^{\ell}.$$

Thus, by applying equation 3.2, we conclude that

$$\#[\mathcal{T}_p] \leq \left(\frac{p+m+\ell-1-|V(\mathcal{G}(\widehat{\mathcal{T}}_p))|}{\ell}\right)^{\ell} \leq \left(\frac{p+m+\ell-1-(2m+\ell-1)}{\ell}\right)^{\ell},$$

and so $\#[\mathcal{T}_p] \leq ((p-m)/\ell)^{\ell}$, which completes the proof. □

We now consider the tightness of the bounds in Theorem 17 by considering some special cases. Notice first that the lower bound is tight exactly when each chordal component is a single edge. The upper bound is tight exactly when $|V(\mathcal{G}(\widehat{\mathcal{T}}_p))| = 2m + \ell - 1$ and each chordal component has exactly $\frac{p-m}{\ell}$ vertices.

Corollary 18. *Suppose $\mathcal{G}(\widehat{\mathcal{T}}_p)$ has precisely one connected component, i.e., $\mathcal{G}(\widehat{\mathcal{T}}_p)$ is a directed tree. Then*

$$2^{\ell} \leq \#[\mathcal{T}_p] \leq \left(\frac{p-1}{\ell}\right)^{\ell},$$

and every directed tree $\mathcal{T}_p$ for which the upper bound is tight has the same subtree $\mathcal{G}(\widehat{\mathcal{T}}_p)$, namely $G_1(\ell)$ with all edges directed inwards.

Proof. The statement of the bounds is immediate from Theorem 17. So we only need to verify the claim on the tightness of the upper bound. It follows from the more general bounds described above, that the upper bound is tight exactly when $|V(\mathcal{G}(\widehat{\mathcal{T}}_p))| = \ell + 1$ and each chordal component has exactly $\frac{p-1}{\ell}$ vertices. Since the chain components $\tau_1, \ldots, \tau_\ell$ are all distinct and $\mathcal{G}(\widehat{\mathcal{T}}_p)$ is a directed tree with $\ell + 1$ vertices, then each τ_j is adjacent to exactly one of the ℓ vertices of $\mathcal{G}(\widehat{\mathcal{T}}_p)$, and there remains only one vertex to connect these ℓ vertices. Therefore, the skeleton of $\mathcal{G}(\widehat{\mathcal{T}}_p)$ is the star $G_1(k+1)$. Moreover, since all essential edges in $\widehat{\mathcal{T}}_p$ are exactly the edges of $\mathcal{G}(\widehat{\mathcal{T}}_p)$, then all edges of $\mathcal{G}(\widehat{\mathcal{T}}_p)$ must be directed inwards towards the center node. An example of a graph for which this upper bound is tight is presented on the left in Figure 15. □

To complement Corollary 18 we next consider the case when $\widehat{\mathcal{T}}_p$ has only one chain component.

Corollary 19. *Suppose $\widehat{\mathcal{T}}_p$ has precisely one chain component τ_1. Then*

$$m \leq \#[\mathcal{T}_p] \leq p - 2m,$$

and both bounds are tight when $\tau_1 = G_1(m-1)$.

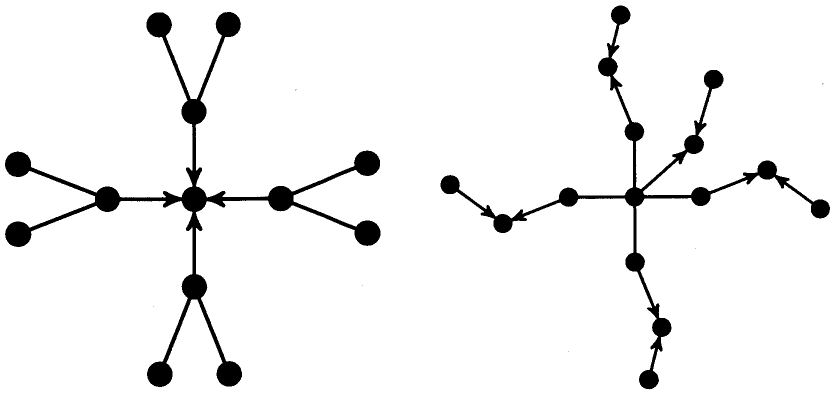


FIGURE 15. Graphs for which the bounds are tight when $m = 1$ (left) and when $k = 1$ (right).

Proof. By Lemma 16 we know that $\#[\mathcal{T}_p] = |V(\tau_1)|$; so the bounds presented here are bounds on the size of the vertex set of the chain component τ_1. Since the connected components $\varepsilon_1, \ldots, \varepsilon_m$ of $\mathcal{G}(\widehat{\mathcal{T}_p})$ are all disjoint, we know that τ_1 contains at least m vertices. On the other hand, since each ε_i contains at least one immorality and attaches to τ_1 at precisely one node, then each ε_i contains at least two nodes that are not also nodes of τ_i. A graph for which the bounds are simultaneously tight is depicted on the right in Figure 15. Notice that the chordal component τ_1 is $G_1(m-1)$. □

Corollary 18 and Corollary 19 suggest the important role of the maximum degree of a graph for the size of MECs. This is further supported and discussed via simulations in Section 5.

4. Immorality Numbers and Star Decompositions

In this section, we focus on the *immorality number* $m(G)$, i.e., the maximum number of immoralities that can exist in a DAG on some underlying undirected graph G. This number is natural to consider when one attempts to enumerate the MECs on G by counting all compatible placements of immoralities. The work done in the previous sections provides the immorality numbers of the following graphs.

(1) The cycle on p vertices: $m(C_p) = \left\lfloor \frac{p}{2} \right\rfloor$,
(2) The path on p vertices: $m(I_p) = \left\lfloor \frac{p-2}{2} \right\rfloor$,
(3) The star on $p+1$ vertices: $m(G_1(p)) = \binom{p}{2}$
(4) The gluing of two stars: $m(G_1(p,q)) = \binom{p+1}{2} + \binom{q}{2}$ when $p \geq q$
(5) The complete bipartite graph $K_{2,p}$: $m(K_{2,p}) = 2\binom{p}{2}$

However, the immorality number $m(G)$ is in general difficult to compute; in fact, as we will see, it is intimately tied to some well-studied NP-complete graph problems from combinatorial optimization. Recall that a *vertex cover* of G is a subset S of vertices of G for which each edge of G is adjacent to some vertex in S. A classic problem in combinatorial optimization is to identify a vertex cover of minimum size for a given graph G. Formally stated, this is the search problem

Problem 20. MINIMUM VERTEX COVER

INPUT: An undirected graph $G = (V, E)$.

OUTPUT: A subset $C \subset V$ such that for all edges $\{u, v\} \in E$ either $u \in C$ or $v \in C$ and $|C|$ is minimized with respect to this property.

The decision version of this problem is called VERTEX COVER [36] and is stated as follows.

Problem 21. VERTEX COVER

INPUT: An undirected graph $G = (V, E)$ and a nonnegative integer k.

PROPERTY: G has a vertex cover of size less than or equal to k.

A search problem Π is said to be *NP-hard* if there is a polynomial time Turing reduction from an NP-complete Π' problem to Π [24, Chapter 5]. That is, if we are given a polynomial time algorithm A for solving Π, then there exists a polynomial time algorithm for solving Π' using A as a hypothetical subroutine. In [47], it is shown that VERTEX COVER is NP-complete even when restricted to triangle-free graphs. Moreover, given a polynomial time algorithm for solving MINIMUM VERTEX COVER, we can certainly derive a polynomial time algorithm to solve VERTEX COVER (for triangle-free graphs or otherwise). Thus, MINIMUM VERTEX COVER is NP-hard for both triangle-free and arbitrary graphs. Analogously, we consider the following search and decision problems related to the computation of the immorality number $m(G)$.

Problem 22. MAXIMUM IMMORALITIES

INPUT: An undirected graph $G = (V, E)$.

OUTPUT: A DAG $\mathcal{G}$ with skeleton G and $m(G)$ immoralities.

Problem 23. IMMORALITES

INPUT: An undirected graph $G = (V, E)$ and a nonnegative integer k.

PROPERTY: There is a DAG $\mathcal{G}$ with skeleton G having at least k immoralities.

In the following, we will identify a polynomial time Turing reduction of MINIMUM VERTEX COVER to MAXIMUM IMMORALITIES when restricted to triangle-free graphs. A polynomial time solution to MAXIMUM IMMORALITIES would trivially yield a polynomial time solution to the same problem in the triangle-free case. Since this would in turn solve an NP-complete problem, we can conclude that the general instance of MAXIMUM IMMORALITIES is NP-hard. This will prove Theorem 1, which was stated in Section 1.

In order to reduce MINIMUM VERTEX COVER to MAXIMUM IMMORALITIES for triangle-free graphs, we will utilize a notion of *star decompositions* of G. In the special cases of the complete bipartite graph $K_{p,p}$ and a family of circulant graphs, we will then use this connection to compute $m(G)$.

4.1. **Star Decompositions.** Let $G = (V, E)$ be a connected, undirected graph. Recall that a *p-star* is the complete bipartite graph $K_{1,n}$ and its *center* is the unique degree p node. A collection of stars $\{S_1, \ldots, S_k\}$ is called a *star decomposition of* G if each S_i is a subgraph of G and each edge of G is an edge of exactly one star in the collection. Our definition of star decomposition is a bit more general than the standard notion studied in graph decompositions. The classic notion of a star decomposition adds the requirement that the stars $S_1, \ldots, S_k$ are all isomorphic to one another. While the literature on which graphs admit a star decomposition of this type is quite extensive [9, 16, 29, 61, 62, 63], there is substantially less work relating to the more general notion we utilize here [41].

In the following, the *trivial star* refers to $K_{1,0}$, and the *size* of a star S is the size of its edge set, which we denote by $|S|$. The *size* of a star decomposition $\mathcal{S}$ is the number of stars in the decomposition, and it is denoted $|\mathcal{S}|$. Given a star decomposition $\mathcal{S} = \{S_1, \ldots, S_k\}$ let $v(\mathcal{S}) \in \mathbb{R}^k$ denote the vector of the sizes of stars in $\mathcal{S}$ ordered greatest-to-least from left-to-right. So if $|S_1| \geq |S_2| \geq \cdots \geq |S_k|$ then $v(\mathcal{S}) = (|S_1|, |S_2|, \ldots, |S_k|)$. If $\mathcal{S}$ is a star decomposition of size k with cardinality vector $v(\mathcal{S}) \in \mathbb{R}^k$, for $m \geq k$ we embed $v(\mathcal{S}) \in \mathbb{R}^m$ by appending zeros to the right end of $v(\mathcal{S}) \in \mathbb{R}^k$. Notice that this corresponds to appending trivial stars to $\mathcal{S}$. We call a star decomposition of G *reduced* if it contains no trivial stars. Notice that the largest reduced

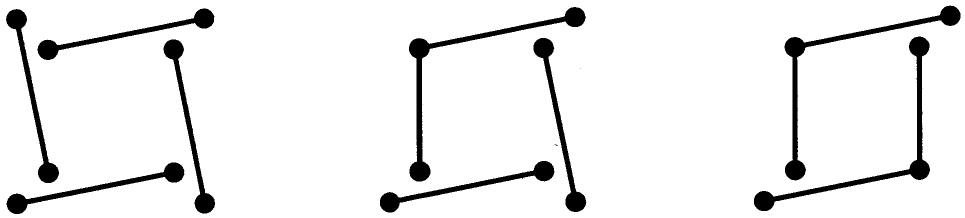


FIGURE 16. The three nonisomorphic reduced star decompositions of the 4-cycle

star decomposition contains at most $|E|$ stars. A *minimum* star decomposition of G contains the minimum number of stars over all star decompositions of G. Notice that a minimum star decomposition will always be reduced. Since the maximum number of stars in a reduced star decomposition of G is $|E|$, then any minimum star decomposition contains at most $|E|$ stars. Also, given a star decomposition $\mathcal{S}$, we call the set of all centers of stars in $\mathcal{S}$ the *center set of* $\mathcal{S}$, and we denote it by $C(\mathcal{S})$. Note that if a star consists only of a single edge, then we simply choose one of its endpoints to be the center node.

For any DAG $\mathcal{G}$ on the undirected graph G we can construct a star decomposition of G as follows. For each node $v \in V$, consider the substar S_v in G whose center is v and whose edges are those directed into v in the DAG $\mathcal{G}$. The *star decomposition of* G *induced by* $\mathcal{G}$ is then

$$\mathcal{S}(\mathcal{G}) := \{S_v : v \in V\}.$$

Notice that an induced star decomposition will not be reduced, and may contain intervals $K_{1,1}$.

Remark 24. Not all star decompositions of a graph G are induced by some DAG on G. For example, any graph has a star decomposition consisting of precisely its set of edges. In the case of the 4-cycle, for instance, this decomposition cannot arise from a DAG.

Since a star decomposition induced by a DAG always contains at least one trivial star, we make the following important definition. A minimum star decomposition of G is *induced* by a DAG $\mathcal{G}$ on G if it is a reduction of the star decomposition induced by $\mathcal{G}$.

Example 25. Consider C_4, the cycle on 4 nodes. Up to isomorphism, C_4 admits the three reduced star decompositions depicted in Figure 16. From this we can see that the minimum star decompositions of C_4 are all isomorphic to $\{K_{1,2}, K_{1,2}\}$. The two right-most star decompositions in Figure 16 are each induced by DAGs. For example, the middle decomposition is induced by the DAG $\mathcal{G}_1$ and

the right-most decomposition is induced by the DAG $\mathcal{G}_2$ as depicted in Figure 17. The left-most star decomposition in Figure 16 is the maximum cardinality reduced star decomposition of the 4-cycle, which consists of exactly one copy of $K_{1,1}$ for each edge of C_4.

Example 25 demonstrates the properties of minimum star decompositions that we will use to study the immorality number of triangle-free graphs. Notice first that the center set of each star decomposition is a vertex cover of C_4 and that the minimum vertex covers of G are center sets of minimum star decompositions. Indeed, there exists a many-to-one correspondence between minimum star decompositions and minimum vertex covers of G.

Lemma 26. *Suppose $\mathcal{S}$ is a minimum star decomposition with center set $C(\mathcal{S})$. Then $C(\mathcal{S})$ is a minimum vertex cover of G.*

Proof. Recall that the center set $C(\mathcal{S})$ of any star decomposition $\mathcal{S}$ is a vertex cover of G. Therefore, any minimum star decomposition has to be at least as large as any minimum vertex cover of G. Suppose that for any minimum vertex cover C of G we can find a star decomposition of G with center set C. Then it follows that any minimum star decomposition has size exactly that of a minimum vertex cover of G. Moreover, the center set of any minimum star decomposition must be a minimum vertex cover. Thus, to complete the proof, we need only show that any minimum vertex cover of G is the center set of some star decomposition of G.

For a node v of a graph G we let $N[v]$ denote the neighbors of v in G including the node v itself. Suppose that $C = \{c_1, \ldots, c_k\}$ is a minimum vertex cover of G. Let $\mathcal{S}(C) = \{S_1, \ldots, S_k\}$ denote the star decomposition of G given by setting

$$S_1 := \langle N[c_1] \rangle,$$

$$S_i := \langle N[c_i] \setminus (\cup_{j \leq i} N[c_j]) \rangle, \text{for } i > 1.$$

FIGURE 17. The DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ and their induced (nonreduced) star decompositions.

Since $\mathcal{S}$ is a star decomposition of G with center set C, this completes the proof. □

Lemma 27. *Suppose C is a minimum vertex cover of G and $\mathcal{S}$ is any star decomposition of G with center set C. Then $\mathcal{S}$ is a minimum star decomposition.*

Proof. Recall that the center set $C(\mathcal{S})$ of any star decomposition $\mathcal{S}$ of G is a vertex cover of G. Thus, just as stated in the proof of Lemma 26, we know that any star decomposition of G is at least as large as any minimum vertex cover of G. By the construction in the proof of Lemma 26, we know in fact that this lower bound is tight. Thus, any star decomposition with center set that is a minimum vertex cover must have minimum size. □

Lemma 28. *Suppose $\mathcal{S}$ is a minimum star decomposition of G with cardinality vector $v(\mathcal{S}) \in \mathbb{R}^{|E|}$ such that $v(\mathcal{S})^T v(\mathcal{S})$ is maximum over all star decompositions of G. Then $\mathcal{S}$ is induced by some DAG with skeleton G.*

Proof. Note first that any star decomposition $\mathcal{S} = \{S_1, \ldots, S_k\}$ of G is induced by some directed, but not necessarily acyclic, graph $\mathcal{G}(\mathcal{S})$. Namely, $\mathcal{G}(\mathcal{S})$ is the directed graph whose arrows are given by directing all edges of S_i so that their heads are at the center node of S_i for all $i \in [k]$. Since each edge of G appears in exactly one star in $\mathcal{S}$, this definition yields a unique directed graph.

For the sake of contradiction, suppose $\mathcal{S}$ is a minimum star decomposition of G for which $v(\mathcal{S})^T v(\mathcal{S})$ is maximized, but $\mathcal{S}$ is not induced by a DAG. Then $\mathcal{S}$ is induced by the directed graph $\mathcal{G}(\mathcal{S})_0 := \mathcal{G}(\mathcal{S})$ constructed in the previous paragraph. By assumption, $\mathcal{G}(\mathcal{S})_0$ contains some directed cycles. Notice that if v is any node contained in a directed cycle, then by the construction of $\mathcal{G}(\mathcal{S})_0$ we know $v \in C(\mathcal{S})$, since v has nonzero indegree in $\mathcal{G}(\mathcal{S})_0$. Therefore, all vertices in all directed cycles in $\mathcal{G}(\mathcal{S})_0$ lie in the center set $C(\mathcal{S})$.

Let v_0 be a node of highest G-degree that is contained in a directed cycle in $\mathcal{G}(\mathcal{S})_0$. Reorient all arrows of $\mathcal{G}(\mathcal{S})_0$ so that v_0 is a sink, and denote the resulting directed graph by $\mathcal{G}(\mathcal{S})_1$. Notice that the directed cycles in $\mathcal{G}(\mathcal{S})_1$ are precisely the directed cycles of $\mathcal{G}(\mathcal{S})_0$ that do not use the node v_0. In particular, $\mathcal{G}(\mathcal{S})_1$ contains strictly less directed cycles than $\mathcal{G}(\mathcal{S})_0$. This is because changing node v_0 into a sink eliminated precisely the directed cycles that passed through v_0.

We iterate this procedure as follows: Let

$$v_i \in V \setminus \bigcup_{j=1}^{i-1} N[v_j]$$

be a node of highest G-degree that is contained in a directed cycle in $\mathcal{G}(\mathcal{S})_i$. Reorient all arrows of $\mathcal{G}(\mathcal{S})_i$ so that v_i is a sink, and denote the resulting directed graph by $\mathcal{G}(\mathcal{S})_{i+1}$. Note that $\mathcal{G}(\mathcal{S})_{i+1}$ contains strictly less directed cycles than $\mathcal{G}(\mathcal{S})_i$. Therefore, iterating this process must result in some DAG $\mathcal{G}(\mathcal{S})_m$.

The center set of the (reduced) induced star decomposition of $\mathcal{G}(\mathcal{S})_m$ is contained in the center set of $\mathcal{S}$, since all nodes on any directed cycle in $\mathcal{G}(\mathcal{S})_0$ are contained in $C(\mathcal{S})$. Therefore, since $\mathcal{S}$ is a minimum star decomposition, so is $\mathcal{S}(\mathcal{G}(\mathcal{S})_m)$.

Now consider the associated vectors $v(\mathcal{S}), v(\mathcal{S}(\mathcal{G}(\mathcal{S})_m)) \in \mathbb{R}^{|E|}$. Since at each step in the construction of $\mathcal{G}(\mathcal{S})_m$ we selected a center node v_i of highest possible degree, then $v(\mathcal{S}) \prec_{\text{lex}} v(\mathcal{S}(\mathcal{G}(\mathcal{S})_m))$, and in particular

$$v(\mathcal{S})^T v(\mathcal{S}) < v(\mathcal{S}(\mathcal{G}(\mathcal{S})_m))^T v(\mathcal{S}(\mathcal{G}(\mathcal{S})_m)),$$

which is a contradiction. □

In some special instances when the minimum star decompositions of a graph G are well-understood, we can use this theory to compute $m(G)$. Recalling Example 25, notice that the minimum star decompositions of C_4 are all isomorphic to one another as forests, and each minimum star decomposition of C_4 is induced by a DAG. With this example in mind, we prove the following theorem.

Theorem 29. *Let G be a triangle-free, undirected graph whose minimum star decompositions are all isomorphic to one another as forests. Then given any minimum star decomposition $\mathcal{S}(G) = \{S_1, \ldots, S_k\}$ of G the immorality number of G is*

$$m(G) = \sum_{i=1}^{k} \binom{|S_i|}{2}.$$

Proof. Since the maximum size of a minimum star decomposition is $|E|$, we can simply assume $k = |E|$ by filling out the set with trivial stars. That is, without loss of generality we assume that all star decompositions considered have the same cardinality $k = |E|$, but may contain trivial stars. A minimum star decomposition is then simply one with the maximum number of trivial stars.

Recall that for every DAG $\mathcal{G}$ on G we can construct the induced star decomposition $\mathcal{S}(\mathcal{G}) = \{S_1, \ldots, S_k\}$. Since G is triangle-free, the number of immoralities in $\mathcal{G}$ is precisely

$$\sum_{i=1}^{k} \binom{|S_i|}{2}.$$

Each such induced star decomposition admits a vector in $\mathbb{R}^k$ for each permutation $\sigma \in S_k$ of cardinalities

$$(|S_{\sigma(1)}|, |S_{\sigma(2)}|, \ldots, |S_{\sigma(k)}|) \in \mathbb{R}^k,$$

and we let $v(\mathcal{G})$ denote any one of these vectors. More generally, any star decomposition $\mathcal{S}$ of G admits such a vector of cardinalities for each permutation $\sigma \in S_k$, any one of which we denote by $v(\mathcal{S}) \in \mathbb{R}^k$. Let $\mathcal{V}(G)$ denote the set of all possible choices of vectors $v(\mathcal{S})$ for all possible star decompositions of G. Then our goal is to maximize the objective function

$$\sum_{i=1}^{k} \binom{x_i}{2} \tag{4.1}$$

over the set $\mathcal{V}(G) \subset \mathbb{Z}^k_{\geq 0}$. Since the objective function satisfies

$$\sum_{i=1}^{k} \binom{x_i}{2} = \frac{1}{2}\left(\sum_{i=1}^{k} x_i^2 - \sum_{i=1}^{k} x_i\right),$$

and for all $(x_1, \ldots, x_k) \in \mathcal{V}(G)$, we have that

$$\sum_{i=1}^{k} x_i = |E| = k,$$

then we are interested in solving the integer optimization problem

$$\begin{array}{ll} \text{maximize} & \mathbf{x}^T\mathbf{x} \\ \text{subject to} & \sum_{i=1}^{k} x_i = k, \\ & \mathbf{x} \in \mathbb{Z}^k_{\geq 0}, \\ & \mathbf{x} \in \mathcal{V}(G). \end{array}$$

The presentation of this optimization problem is redundant, but it is to emphasize the fact that any vector in $\mathcal{V}(G)$ lies in the k^{th} dilate of the probability simplex Δ_k, which we denote by $k\Delta_k$. Therefore, we are simply maximizing the length over all vectors in the probability simplex that also lie in the set $\mathcal{V}(G)$. Since the value of $\mathbf{x}^T\mathbf{x}$ strictly increases as we approach the boundary of $k\Delta_k$ then the star decompositions with the maximum number of trivial stars will yield the maximum value of the objective function. These are the minimum star decompositions, all of which are isomorphic as trees, and therefore have the same vectors $v(\mathcal{S})$ up to a permutation of coordinates. Since we have assumed that at least one of these star decompositions is induced by a DAG, it

follows that the maximum value of the original objective function (4.1) is the immorality number of G. □

Collectively, Lemmas 26, 28, and Theorem 29 allow us to prove Theorem 1.

4.2. **Proof of Theorem 1.** Let G be a triangle-free graph, and suppose that we have a polynomial time algorithm that returns a DAG $\mathcal{G}^*$ with skeleton G for which $\mathcal{G}^*$ has the maximum number of immoralities. By Lemma 28 and Theorem 29, we know that the maximum value of $\sum_{i=1}^{|E|} \binom{|S_i|}{2}$ is achieved by a minimum star decomposition induced by a DAG. Since the value of $\sum_{i=1}^{|E|} \binom{|S_i|}{2}$ is exactly equal to the number of immoralities in a DAG with a triangle-free skeleton, it follows that our DAG $\mathcal{G}^*$ induces a minimum star decomposition $\mathcal{S}(\mathcal{G}^*)$ that maximizes $\sum_{i=1}^{|E|} \binom{|S_i|}{2}$. We know by Lemma 26 that the center set $C(\mathcal{S}(\mathcal{G}^*))$ is a minimum vertex cover of G. Therefore, we have a polynomial time algorithm for computing a minimum vertex cover of the triangle-free graph G. It is clear that a polynomial time algorithm for MAXIMUM IMMORALITIES for arbitrary graphs trivially yields a polynomial time algorithm for MAXIMUM IMMORALITIES for triangle-free graphs. Therefore, since MINIMUM VERTEX COVER is NP-complete for triangle-free graphs, we know that the general instance of MAXIMUM IMMORALITIES is NP-hard. This completes the proof of Theorem 1. □

Remark 30. Recall that there is trivially a polynomial time Turing reduction of MINIMUM VERTEX COVER to VERTEX COVER. Conversely, it is well-known that VERTEX COVER is *self-reducible*. That is, given a polynomial time algorithm for VERTEX COVER one can find a polynomial time algorithm solving MINIMUM VERTEX COVER. Collectively, this says that solving MINIMUM VERTEX COVER is no more or no less hard than solving VERTEX COVER. Since the former direction is trivial, the critical observation made here is the self-reducibility of VERTEX COVER.

The proof of self-reducibility for VERTEX COVER is standard across many NP-complete structural search problems for graphs, and it goes as follows. Given a graph $G = (V, E)$, the minimum size of a vertex cover must be between 0 and $|V|$. Thus, by a binary search, we can determine in polynomial time the size k^* of a minimum vertex cover of G. Then, to recover a vertex cover C with size k^* of G, we first pick a vertex v and delete it from G. If the resulting graph has a vertex cover of size $k^* - 1$, then v is in a minimum vertex cover of G, if not we return the vertex v, and

repeat with another vertex. Iterating this procedure produces a minimum vertex cover of G in polynomial time.

While the self-reducibility of many other graph structure search problems are proved using a similar argument, this proof is unusable for IMMORALITIES. The analogous argument for IMMORALITIES would require considering all subsets of neighbors of the node v and deleting the corresponding star. Since the number of such queries for a given vertex v is only bounded by $\binom{\deg(v)}{2}$, this algorithm is not polynomial in time. However, this does not prove that IMMORALITIES is not self-reducible, nor does it prove that IMMORALITIES is not NP-complete.

We now present a few special cases in which star decompositions allow us to compute $m(G)$ via an application of Theorem 29: In Section 4.3 we compute $m(G)$ for the complete bipartite graph $K_{p,p}$ and in Section 4.4 for some special circulant graphs.

4.3. **The complete bipartite graph $K_{p,p}$.** In [26] Gillespie and Perlman note that the maximum number of induced 3-paths over all skeletons on n nodes, for each $n \leq 10$ is given by the complete bipartite graph $K_{\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil}$. The number of induced 3-paths in the graph $K_{\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil}$ is quickly seen to be

$$a_n = \left\lfloor \frac{n}{2} \right\rfloor \left\lceil \frac{n}{2} \right\rceil \frac{n-2}{2},$$

which is sequence A111384 of [56]. Since induced 3-paths in an undirected graph G are exactly the possible locations of immoralities in a DAG with skeleton G, it is reasonable to ask for the immorality number of the complete bipartite graph $K_{p,p}$. As one would hope, the immorality number of $K_{p,p}$ turns out to be exactly one half the number of induced 3-paths.

We now use Theorem 29 to compute the immorality number of $K_{p,p}$ via star decompositions. To do so, we make one additional observation.

Lemma 31. *The minimum star decompositions of $K_{p,p}$ are all isomorphic to*

$$\left\{ \underbrace{K_{1,p}, K_{1,p}, \ldots, K_{1,p}}_{p \text{ times}} \right\}.$$

Proof. We prove a slightly stronger statement. Let $N[v]$ denote the subgraph of a graph G induced by the vertex v and its set of neighbors. Let the vertices of $K_{p,p}$ be the partitioned set $A \sqcup B$ where $A := \{a_1, \ldots, a_p\}$ and $B := \{b_1, \ldots, b_p\}$. We claim that the minimum star decompositions of $K_{p,p}$

are only

$$\{N[a_i] : i \in [p]\} \qquad \text{and} \qquad \{N[b_i] : i \in [p]\}.$$

To see this assume otherwise. Suppose that $\{S_1, \ldots, S_k\}$ is a minimum star decomposition of $K_{p,p}$, and let c_i denote the center of star S_i for all $i \in [k]$. We also set $C := \{c_1, \ldots, c_k\}$.

Suppose first that $k = p$ and that $A \cap C \neq \emptyset$ and $B \cap C \neq \emptyset$. Without loss of generality, assume that $A \cap C = \{a_1, \ldots, a_\ell\}$ for some $\ell < k$. Then for all $i > \ell$ it must be that $b_i \in B \cap C$, since otherwise the edge $\{a_i, b_i\}$ would not appear in any star in $\{S_1, \ldots, S_k\}$. Since $k = p$, it follows that $B \cap C = \{b_{\ell+1}, \ldots, b_p\}$. However, this means that for all $i \leq \ell$ and $j \geq \ell + 1$, the edges $\{a_j, b_i\}$ are not in any star, which is a contradiction.

Now suppose that $k < p$. It is quick to see that if $C \subset A$ or $C \subset B$ then there exist edges of $K_{p,p}$ not contained in stars. So $A \cap C \neq \emptyset$. The proof then follows from applying the same argument as in the case when $k = p$ to derive a contradiction. □

Theorem 32. *The immorality number of $K_{p,p}$ is $p\binom{p}{2}$.*

Proof. Notice that $K_{p,p}$ is triangle-free. By Lemma 31, the minimum star decompositions of $K_{p,p}$ are all isomorphic as forests. Moreover, any such minimum star decomposition is induced by a DAG $\mathcal{G}$ on $K_{p,p}$ that has exactly p sinks located along either the node set A or the node set B. The result then follows from Theorem 29. □

4.4. **Some triangle-free circulants.** Circulant graphs are natural generalizations of the cycle graphs, and both their independence polynomials and independence numbers have been studied extensively [4, 7, 8, 10, 32]. However, as is shown by these references, there is no known general formula for the Fibonacci number, the independence number, nor the independence polynomial of these graphs. As is exhibited by the various examples of trees studied in Section 3, it appears that graphs with more high degree nodes will consistently admit MECs of smaller size. This makes d-regular graphs a fertile testing ground for the enumeration of MECs, and their associated sizes. Since circulants are always d-regular, we wish to study their MEC sizes and quantities as distributed by number of immoralities analogously to the case of independence polynomials. Similar to the case of independent sets, this picture is difficult to achieve, even in special cases. In the following, we compute the immorality number of some special classes of triangle-free circulants.

Recall that a circulant on p nodes is a graph whose nodes are identified with $\mathbb{Z}/p\mathbb{Z}$, and whose edges are given by a specified *connection set* $C \subset \mathbb{Z}/p\mathbb{Z}$. In the undirected setting, we assume

C is closed under additive inverses. The circulant on p nodes with connection set C is denoted $X(p, C)$ and has edges $\{i, j\}$ for all pairs i and j satisfying $i - j \in C$. We often abbreviate the connection set C via a subset of $\left[\frac{p}{2}\right]$ by omitting the additive inverse of each element. As a corollary to Theorem 29, we can determine the immorality number of some triangle-free circulant graphs.

Corollary 33. *Let p be even, and suppose that $X(p, C)$ is a triangle-free circulant graph containing a p-cycle for which the maximum independent subset is of size $\frac{p}{2}$. Then*

$$m(X(p, C)) = \frac{p}{2}\binom{2|C|}{2}.$$

Proof. Recall that a set of nodes in a graph G is a minimum vertex cover if and only if its complement is an independent set in G. Since $X(p, C)$ contains a p-cycle, then without loss of generality we can assume that $1 \in C$. Since $1 \in C$ and the maximum independent subset of $X(p, C)$ is equal to the one of C_p of size $p/2$, then any minimum independent set is given by selecting precisely every other vertex of the graph as we walk along the p-cycle given by $1 \in C$. Moreover, such a vertex set is also a minimum vertex cover. Thus, if $\{c_1, \ldots, c_{\frac{p}{2}}\}$ is a maximum independent set in $X(p, C)$, then there is only one possible star decomposition with center set $\{c_1, \ldots, c_{\frac{p}{2}}\}$, namely

$$\{\langle N[c_1]\rangle, \ldots, \langle N[c_{\frac{p}{2}}]\rangle\} \simeq \{K_{1,2|C|}, K_{1,2|C|}, \ldots, K_{1,2|C|}\}.$$

Since the only two maximum independent sets in $X(p, C)$ share this property, it follows from Lemma 27 that all minimum star decompositions of G are isomorphic. Thus, by Theorem 29 we conclude that

$$m(X(p, C)) = \frac{p}{2}\binom{2|C|}{2}.$$

□

Notice that Corollary 33 applies to any triangle-free circulant with p even, $1 \in C$, which has all other elements of C being odd. On the other hand, we cannot apply the same techniques to compute the immorality numbers for p odd, since such circulants may contain nonisomorphic minimum star decompositions.

Remark 34. Notice that we can use Theorem 29 to compute some of the immorality numbers listed at the start of this section very quickly. For example, the minimum star decomposition of $G_1(p)$ is clearly the graph itself, so by Theorem 29 $m(G_1(p)) = \binom{p}{2}$. Similarly, the graph $K_{2,p}$ can be

decomposed into two stars, both of which are isomorphic to $K_{1,p}$, in precisely one way, and therefore this is its unique minimum star decomposition. It follows by Theorem 29 that $m(K_{2,p}) = 2\binom{p}{2}$. Similarly, Theorem 29 implies that $m(G_2(p,p)) = \binom{p+1}{2} + \binom{p}{2}$.

On the other hand, if $p \neq q$, then the minimum star decompositions of $G_2(p,q)$ are all size two but need not be isomorphic. Therefore, Theorem 29 does not apply. In a similar fashion, Theorem 29 cannot be applied to the caterpillar graphs W_p.

5. Computational Analysis

In this section, we describe the computer program we used to test our conjectures and collect relevant statistics. This program can be found at `https://github.com/aradha/mec_generation_tool`. Using the data collected, we then empirically examined how the structure of the skeleton effects the size and number of its associated Markov equivalence classes. In particular, we examined how the size and number of Markov equivalence classes relates to the clustering coefficient and distribution of high degree nodes in the underlying skeleton for general and triangle-free graphs.

The theory developed in this paper was motivated by the first computer program written for the enumeration of MECs. This program was created by Gillespie and Perlman who described their program, and the resulting data, in [26]. For each skeleton on $p \leq 10$ nodes, the Gillespie and Perlman algorithm logged the maximum number of induced 3-paths, the maximum number of MECs, the total number of equivalence classes, and the size of each class. Our program expands on this original program in two ways: for skeletons on $p \leq 10$ nodes, our program collects more data about each skeleton, and it produces all such data for all triangle-free skeletons on $p \leq 12$ nodes. The new program now catalogues the same information as the original Gillespie and Perlman algorithm for each skeleton as well as the degree sequence of the skeleton, the number of triangles, and the number of immoralities per equivalence class. This additional data, especially in the triangle-free setting, allows us to more carefully analyze how the structure of the skeleton impacts the number and size of its associated MECs. In the following, we first provide a brief description of the algorithm and the hashing scheme used. We then validate the correctness of implementation of the algorithm. Finally, we analyze the data gathered. We first validate Theorem 2 and we also discuss the analogous result in the case of unconnected graphs. Then we compare the size and number of Markov equivalence classes to the clustering coefficient, the average degree, and the maximum degree of the skeleton. We also study how Markov equivalence class size relates to

the ratio of immoralities in the essential graph of the class to the number of induced 3-paths in the underlying skeleton. All analyses are conducted for both general (connected) and triangle-free graphs.

5.1. **The algorithm.** There are three main components in our program's data pipeline which we now describe.

(1) The first component is the main class that reads in skeleton data generated using tools from `nauty` and `Traces` [43].

(2) The second component is a DAG generator that directly generates all DAGs on a given skeleton. Such a generator is realized using the algorithm published by Barbosa and Swarcfiter in [3]. It is essential to directly generate all DAGs rather than generating all directed graphs and then pruning out the ones containing cycles, since the number of directed cyclic graphs dominates the number of directed graphs for a large number of vertices.

(3) The final main component is a DAG enumerator that generates the frequency vector $M(G)_{\text{freq}}$ when given the DAGs on a given skeleton G. In order to generate the number of MECs of each size on a given skeleton, this component creates a bit representation for each MEC by first creating a bit mask of the possible immoralities that could occur in the skeleton. Each DAG is then traversed. If three vertices are found to be in an immorality then the Cantor pairing function is used to hash the triple of their integer labels to the location of the bit in the immorality bit mask. Since the Cantor pairing function is invertible and since the number of vertices in each graph is small, we have a valid, non-overflowing hash function. After comparing the resulting hashes for all DAGs on the given skeleton, a pair of integers is returned for each MEC: the number of immoralities in the MEC and the size of the MEC.

It is an important feature of the algorithm that this component of the pipeline has access to data on the given skeleton. This allows us to collect data on the skeleton in relation to each MEC. Using this, for each skeleton we record the number of induced 3-paths, the degree sequence, the number of edges, and the number of triangles.

To handle the around 12 million undirected graphs on 10 nodes, we split these graphs into approximately 500 files across 10 directories, allocating 16 threads to process each directory. Running this process in parallel takes 5 days as compared to the 253 CPU hours (approximately 94 days) by Gillispie and Perlman [25].

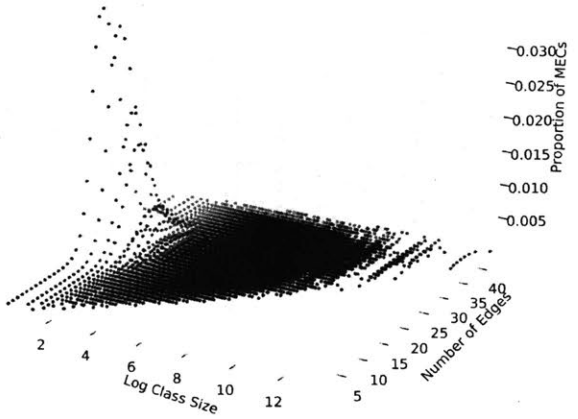


FIGURE 18. The proportion of MECs on connected graphs with 10 nodes as distributed by log class size and number of edges.

5.2. **Correctness of the algorithm.** In order to verify the correctness of our implementation, we matched our program's output with that of the algorithm used by Gillespie and Perlman. In Figure 18 for instance, we are able to use our program to reproduce the same distribution of the proportion of MECs with respect to class size and number of edges as observed in [26, Figure 4]. We also compared performance in terms of speed and memory utilization. Our program runs in nearly the exact time measured by Gillespie and Perlman: we measured that our algorithm also takes around three minutes for eight vertices and only a few seconds or milliseconds for fewer vertices.

However, we have better memory utilization than Gillespie and Perlman as the number of bits we store for hashes is dependent on the number of possible immoralities in the skeleton rather than on the number of possible triples of vertices. We also use Java to do our data processing. Thus, since we only need a print out of the data collected in subsection 5.1 (3) for each skeleton processed, the garbage collector clears out our hash map allocation after each skeleton. This allows us to not only log the class size and the number of MECs per skeleton, but also the number of immoralities per class as well as the number of induced 3-paths, the degree sequence, the number of edges, and the number of triangles. Thus, despite the fact that our algorithm only matches the Gillespie and Perlman algorithm in time, it is collecting significantly more data per skeleton.

5.3. **Data analysis.** Using the data collected from our program, we first verify that Theorem 2 holds. We then examine a series of plots that highlight the relationships between the structure

$M(G)_{\text{freq}}$	Skeleton 1	Skeleton 2
24	$K_4 \cup K_1 \cup K_1 \cup K_1 \cup K_1 \cup K_1 \cup K_1$	$K_3 \cup K_2 \cup K_2 \cup K_1 \cup K_1 \cup K_1$
48	$K_4 \cup K_2 \cup K_1 \cup K_1 \cup K_1 \cup K_1$	$K_3 \cup K_2 \cup K_2 \cup K_2 \cup K_1$
144	$C_4 \cup K_3 \cup K_1 \cup K_1 \cup K_1$	$K_3 \cup K_3 \cup K_2 \cup K_2$
720	$K_6 \cup K_1 \cup K_1 \cup K_1 \cup K_1$	$K_5 \cup K_3 \cup K_1 \cup K_1$
1440	$K_6 \cup K_2 \cup K_1 \cup K_1$	$K_5 \cup K_3 \cup K_2$
2880	$K_6 \cup K_2 \cup K_2$	$K_5 \cup K_4 \cup K_1$
(72, 24)	$K_4 \cup I_3 \cup K_1 \cup K_1 \cup K_1$	$K_3 \cup I_3 \cup K_2 \cup K_2$

TABLE 1. Table describing the 10-node graphs with the same MEC frequency vector.

of the underlying skeleton and the number and size of the associated MECs. All plots have been generated using `matplotlib` [34].

5.3.1. *Validity of Theorem 2.* After running the algorithm on all connected graphs with up to ten nodes, we verified that there was no pair of skeleta with $p \leq 10$ nodes that have the same frequency vector $M(G)_{\text{freq}}$. This indicates that the MEC frequency vectors $M(G)_{\text{freq}}$ bijectively map to skeletons of connected graphs up to ten nodes. Similarly, when we ran our algorithm on all graphs with ten nodes including graphs that were not necessarily connected, we found that the only collisions occurred on graphs G and H with the following property: Let $G = G_1 \sqcup \cdots \sqcup G_m$ and $H = H_1 \sqcup \cdots \sqcup H_n$ be the decompositions of G and H into connected components. Let $G \cap H$ denote the set consisting of the connected components that are shared between G and H up to isomorphism. Now let $G \setminus G \cap H = G_{i_1} \sqcup \cdots \sqcup G_{i_m}$ and $H \setminus G \cap H = H_{j_1} \sqcup \cdots \sqcup H_{j_n}$, where $i_1, \ldots, i_m \in [m]$ and $j_1, \ldots, j_n \in [n]$, be the remaining subgraphs. Then

$$\prod_{k=1}^{m} |G_{i_k}| = \prod_{\ell=1}^{n} |H_{j_\ell}|.$$

For example, over all graphs with ten nodes, there are seven such examples that occurred. These are shown in Table 1.

5.3.2. *Skeletal structure in relation to the number and size of MECs.* The theory developed in this paper and the computational verification of Theorem 2 support the intuition that the undirected structure of the skeleton of a DAG plays a fundamental role in the size of its MEC as well as the number of other MECs that have the same skeleton. In the triangle-free setting, the results of Section 3 and Theorem 29 highlight the significance of the number and density of high degree nodes within the skeleton of the DAG. In this section, we parse these observations in terms of the

computational data collected via our computer program. Specifically, we will compare class size and the number of MECs per skeleton to skeletal features including average degree, max degree, clustering coefficient, and the ratio of number of immoralities in the essential graph of the MEC to the number of induced 3-paths in the skeleton.

Recall that the *(global) clustering coefficient* of a graph G is defined as the ratio of the number of triangles in G to the number of connected triples of vertices in G. The clustering coefficient serves as a measure of how much the nodes in G cluster together. Figure 19 presents two plots: one compares the clustering coefficient to the log average class size and the other compares it to the average number of MECs. This data is taken over all connected graphs on $p \leq 10$ nodes with 25 edges (to achieve a large number of MECs). As we can see, the average class size grows as the clustering coefficient increases. This is to be expected, since an increase in the number of triangles within the DAG should correspond to an increase in the size of the chordal components of the essential graph. On the other hand, the average number of MECs decreases with respect to the clustering coefficient, which is to be expected given that the class sizes are increasing. This decrease in the average number of MECs empirically captures the intuition that having many triangles in a graph results in fewer induced 3-paths, which represent the possible choices for distinct MECs with the same skeleton.

Figure 20 presents a pair of plots, the first of which compares the average degree of the underlying skeleton of the DAG to the log average class size of the associated MEC. The second plot compares the average degree of the skeleton to the average number of MECs it supports. Both plots present

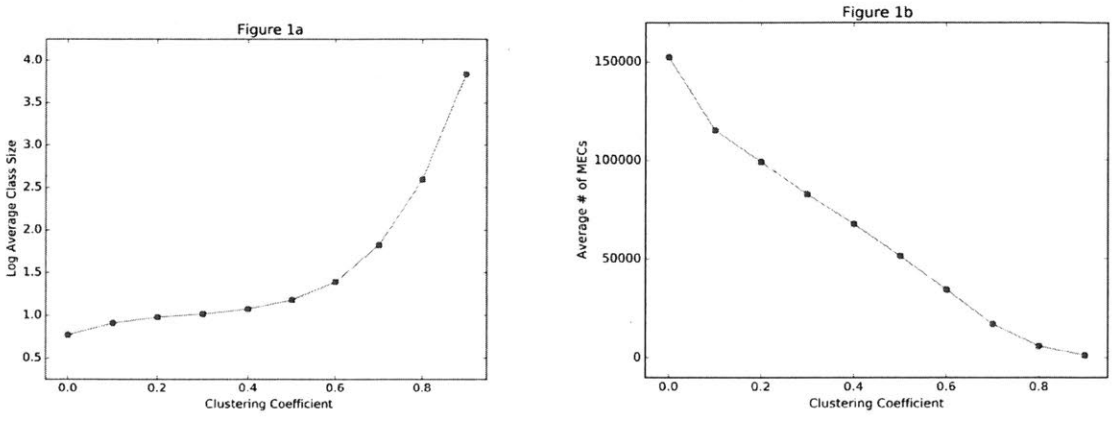


FIGURE 19. Clustering coefficient as compared to log average class size and the average number of MECs.

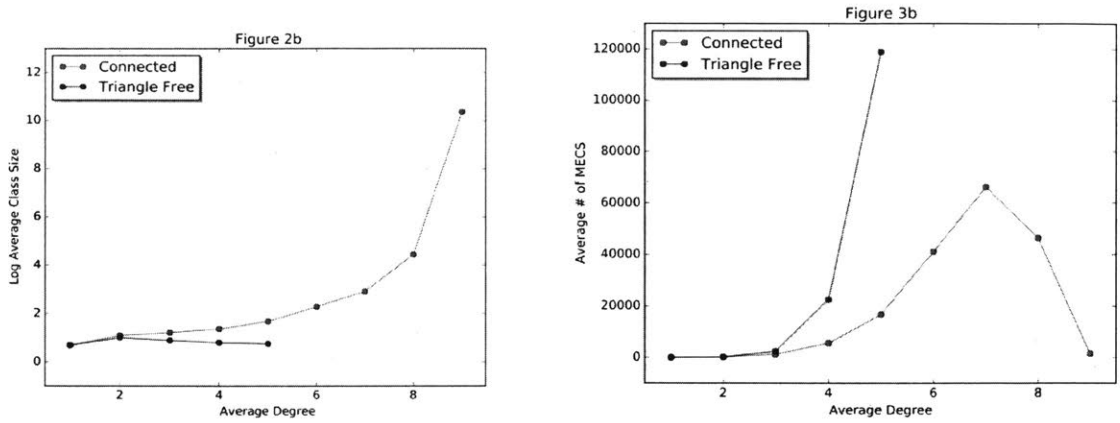


FIGURE 20. Average degree versus log average class size and average number of MECs for all graphs and triangle-free graphs on 10 nodes.

one curve for all connected graphs and a second curve for triangle-free graphs on 10 nodes. For connected graphs on 10 nodes the left-most plot shows a strict increase in the log average MEC class size as the average degree of the nodes in the underlying skeleton increases. This is to be expected since graphs with a higher average degree are more likely to contain larger chordal components. On the other hand, the average class size for triangle-free graphs increases for average degree up until approximately 2.0, and then shows a steady decrease for larger average degree. Since the average degree of a tree on p nodes is $2 - \frac{2}{p}$, this suggests that the largest MECs amongst triangle-free graphs have skeleta being trees. As such, the bounds developed in Section 3 of this paper can be, heuristically, thought to apply more generally to all triangle-free graphs.

The right-most plot in Figure 20 describes the relationship between average degree and the average number of MECs for all connected graphs and triangle-free graphs on 10 nodes. We see from this that in the setting of all connected graphs, the skeleta with the largest average number of MECs appear to have average degree 7, whereas in the triangle-free setting, the higher the average degree the more equivalence classes the skeleta can support. This supports the intuition that the more high degree nodes there are in a triangle-free graph the more equivalence classes the graph can support. Theoretically, a portion of this intuition is captured by Theorem 29. The left-most plot in Figure 21 depicts the relationship between the maximum degree of a node in a skeleton and the average class size on the skeleton for all connected graphs and for triangle-free graphs on at most 10 nodes. For all graphs, the relationship appears to be almost linear beginning with maximum degree 5, suggesting that average class size grows linearly with the maximum degree of

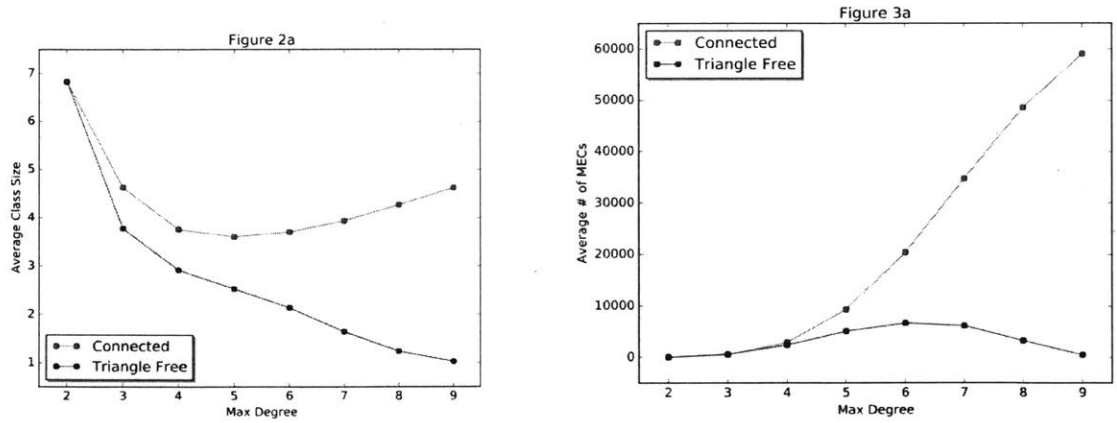


FIGURE 21. Maximum degree versus log average class size and average number of MECs for all graphs and triangle-free graphs on 10 nodes.

the underlying skeleton. This growth in class size is due to the introduction of many triangles as the maximum degree grows. On the other hand, in the triangle-free setting we actually see a decrease in average class size as the maximum degree grows, which empirically reinforces this intuition.

The right-most plot in Figure 21 records the relationship between the maximum degree of a node in a skeleton and the average number of MECs supported by that skeleton for all connected graphs and triangle-free graphs on at most 10 nodes. For all graphs, we see that the average number of MECs grows with the maximum degree of the graphs, and this growth is approximately exponential. In the triangle-free setting, the average number of MECs appears to be unimodal, but would be increasing if we considered also all graphs on $p > 10$. For triangle-free graphs there is only one graph with maximum degree 9, namely the star $G_1(9)$, where the number of MECs is $2^9 - 9$. For connected graphs the average number of MECs is pushed up by those cases consisting of a complete bipartite graph where in addition one node is connected to all other nodes.

The final plot of interest is in Figure 22, and it shows the relationship between Markov equivalence class size and the ratio of the number of immoralities in the essential graph to the number of induced 3-paths in the skeleton for all connected graphs and triangle-free graphs on 10 nodes. That is, it shows the relationship between the class size and how many of the potential immoralities presented by the skeleton are used by the class. It is interesting to note that, in the triangle-free setting, as the class size grows, this ratio appears to approach 0.3, suggesting that most large MECs use about a third of the possible immoralities in triangle-free graphs. In the connected graph setting, as the class size grows, we see a steady decrease in the value of this ratio. This supports the intuition

that a larger class size corresponds to an essential graph with large chordal components and few immoralities.

6. Introduction to Interpretable Neural Networks

There have been significant improvements to feed-forward convolutional networks culminating in the recent success of ResNet [31] in the ImageNet [53] challenge. These (now standard) networks are powerful due to their ability to exploit global context to make a more informed decision. By incorporating non-linear layers, these models use complex combinations of features to derive an accurate label estimate given all the pixels in the input image. The complex interactions between features make it difficult to inspect the learned features visually. Without an easy means of interpreting learned features, the applicability of these models to prediction sensitive areas such as health care is limited. For example, Caruana et al. [11] analyzed models trained to predict the risk of death by pneumonia for patients. They found that these models had learned that having asthma was indicative of a low risk of dying by pneumonia. However, the reason for this learned feature was that patients with asthma were immediately admitted to the ICU when they had pneumonia, and so they would be treated immediately. Thus, asthmatic patients appeared in the training data as examples of patients with low-risk of death by pneumonia. Although often highly accurate, due to the uninterpretability of the learned features, neural networks are often deemed too risky for applications in health care [17].

The goal of our work is to provide a convolutional network for binary classification problems on images that can provide interpretable visual representations of the learned features as heatmaps,

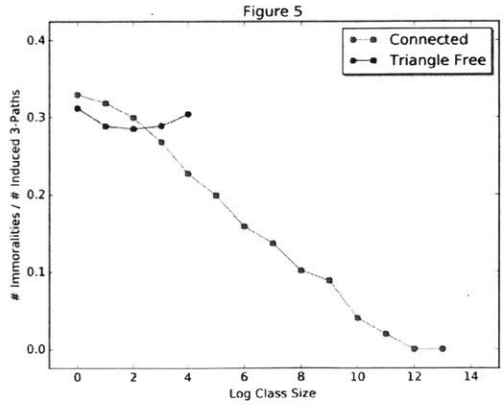


FIGURE 22. Class size versus the ratio of the number of immoralities to the number of induced 3-paths in the essential graph.

simply by inspecting the outputs of the feed-forward layers. To accomplish this, our network *PatchNet* first provides classification decisions on small patches of an image to determine whether a given patch contains features of either class, and then averages the decisions made on all the patches across an image to make a global classification decision. Hence, PatchNet allows for a trade-off between generalization error and feature interpretability: by restricting the size of patches, generalization error increases since the model is limited in global context for classification, but feature interpretability increases as we can visualize the learned features for each patch of an image.

PatchNet is motivated by mean-field approximation techniques from variational inference as well as ensemble methods. As is done in mean-field approximations, instead of learning conditional probability distributions for predicting the label for an image given the entire image, we instead learn a simpler conditional probability distribution for predicting the label given a small patch of the original image. Now unlike a true mean-field approximation, we do not multiply the predictions for each patch to get a global prediction, but rather treat each of these patch predictions as an ensemble of smaller classifiers and average their classifications to generate a global classification decision.

The remainder of this paper is organized as follows. Section 7 describes previous methods for interpreting or rationalizing features learned by neural networks. Section 8 describes the PatchNet architecture, its convergence behavior, how to extract heatmaps for interpreting the learned features, and *Precision-PatchNet*, an extension of PatchNet for classification tasks where obtaining a high precision is important, as is common when working with medical images. We present the results of PatchNet for classifying textures from the Describable Textures Dataset (DTD) [13] and the results of Precision-PatchNet on the ISIC-ISBI Melanoma Classification Challenge [28] in Section 9. We end with concluding remarks and an outline of future work in Section 10.

7. Related Work

A prominent approach used to understand the inner workings of a complex neural network is the deconv technique. Variations of this technique such as DeconvNet [69] or Guided Backpropagation [58] pick a neuron in a convolutional neural network (CNN), do a forward pass on a sample image, and set the chosen neuron's gradient to 1 and all other gradients in the same layer to 0. Doing backpropagation on this signal and blocking the negative gradients, results in an alteration

to the image that represents the neuron's positive contributions to the network as a whole. A similar method was described in [54], where the authors propose an optimization technique that feeds an image forward through a CNN and then backpropagates from the last layer with a specific class target.

Although these techniques allow for visual introspection of the learned features and classes, there are a few drawbacks: (1) All techniques require applying alterations to the true gradient by blocking signals or using regularization parameters. (2) The backpropagation of the gradient through multiple layers provides noisy estimates of the true learned features. (3) The learned features are based on global context, making it inherently difficult to understand how different features work together to create a classification decision and substantially reducing feature interpretability.

A different approach is taken in [39], where a Natural Language Processing model is developed that provides rationales for why a text review is classified as positive or negative. Their model consists of two networks, one to provide tags for whether a word in the review is indicative of "sentiment", and another to classify based on the tagged words. Unfortunately, this approach cannot directly be applied to image classification, since there is no clear analogy between tagging words in a sentence and tagging pixels in an image, as the space of words is much larger than the space of pixel values. Furthermore, naively dropping pixels by techniques such as zeroing them out is not equivalent to dropping irrelevant words in a sentence, since zero value pixels still provide context to the image.

8. PatchNet

In this section, we provide the mathematical motivation for our model architecture, present the architecture, and develop insights into its convergence behavior for special classes of data. We then present a slight modification that incentivizes our model to reduce the false positive rate for classification. We conclude the section with an explanation on how to easily extract global and filter heatmaps from our model to view the learned features.

8.1. Motivation and Notation.

Suppose we are given a dataset $\mathcal{D}$ consisting of a list of images $I^{(1)}, I^{(2)}, \dots I^{(k)}$ with each $I^{(j)} \in \{0, 1, 2 \dots N-1\}^{m \times n \times c}$ along with a corresponding list of labels $y_{I^{(1)}}, y_{I^{(2)}}, \dots, y_{I^{(k)}}$ with each $y_{I^{(j)}} \in \{0, 1\}$. Feed-forward CNNs such as VGG [55] or ResNet [31] directly estimate the distributions $\mathbb{P}_{y|I}(y|I)$, which are conditional distributions given *all* pixel values. Since we will concentrate solely on binary classification, it suffices to estimate $\mathbb{P}_{1|I}(1|I)$.

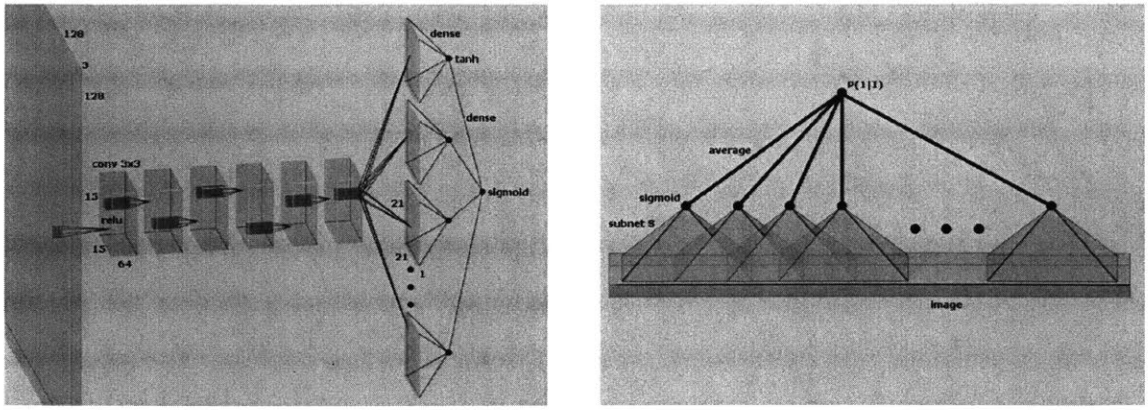


FIGURE 23. PatchNet architecture. The subnet $\mathcal{S}$ is displayed on the left, and the global network consisting of repeated applications of the subnetwork is shown on the right.

Instead of directly estimating the conditional probability distribution given all pixel values in the image, our model estimates conditional probability distributions of patches of pixels in the image and then averages all estimates across the patches to create a global estimate. Formally, if an image is chunked into l possibly overlapping patches $P^{(1)}, P^{(2)}, \ldots, P^{(l)}$ with $P^{(j)} \in \{0, 1, 2, \ldots N-1\}^{m' \times n' \times c}$ with $3 \leq m' \leq m$, $3 \leq n' \leq n$, then our model estimates $\mathbb{P}_{1|I}(1|I)$ as $\mathbb{P}_{1|I}(1|I) = \frac{1}{l}\sum_{j=1}^{l} \mathbb{Q}_{1|P^{(j)}}(1|P^{(j)})$, where $\mathbb{Q}$ is a single learned distribution applied to each patch.

There is an inherent tradeoff between patch size (m', n'), generalization error, and visual interpretability of the learned features. For instance, with $m' = m$ and $n' = n$, our model can mimic any CNN by simply letting the estimate for $\mathbb{Q}$ be the estimate determined by the CNN. In this case, the generalization error achieved by the model is the same as that of the mimicked network, but the visual interpretability of the learned features suffers due to the scale of feature detection. By using small m', n', the distribution $\mathbb{Q}$ is estimated across a smaller input space allowing detection of local features. As we will show in Section 8.3, smaller patch sizes provide visual interpretability of the features used for classification at the expense of a potentially larger generalization error.

8.2. **Model Architecture.** Our model consists of two components: (1) a global network $\mathcal{G}$ that outputs a global classification decision given an entire image; (2) a local network $\mathcal{S}$ that outputs a local classification decision given a patch; see Figure 23.

We now describe the feed-forward pass of our architecture. When an input image I is fed into $\mathcal{G}$, $\mathcal{G}$ chunks the image into a list of l patches of size $m' \times n' \times c$, namely $\left[P^{(1)}, P^{(2)}, \ldots P^{(l)}\right]$ with each $P^{(j)} \in \{0, 1, \ldots 255\}^{m' \times n' \times c}$. Next, these patches are aggregated and sent to $\mathcal{S}$ as a mini-batch.

Then $\mathcal{S}([P^{(1)}, P^{(2)}, \dots P^{(l)}]) = \left[\mathbb{Q}_{1|P^{(1)}}(1|P^{(1)}), \mathbb{Q}_{1|P^{(2)}}(1|P^{(2)}), \dots \mathbb{Q}_{1|P^{(l)}}(1|P^{(l)})\right]$, where $\mathbb{Q}_{1|P^{(j)}}$ is determined as follows. A CNN consisting of 7 layers of convolutions with 64 filters with kernel size 3 and 1 pixel of padding followed by ReLU activations in each layer is applied to $P^{(j)}$ to get 64 $m' \times n'$ filter output images. Then 64 linear models are applied as a dot product to each of these filter output images to reduce the $m' \times n'$ outputs to size 1 outputs, and a Tanh activation is applied to each output. Lastly, a linear layer is applied as a dot product to these 64 outputs resulting in a single value $\tilde{\mathbb{Q}}_{1|P^{(j)}}$ for each patch, which is converted to $\mathbb{Q}_{1|P^{(j)}}$ by a sigmoid transformation. Finally, $\mathcal{G}$ averages $\left[\mathbb{Q}_{1|P^{(1)}}(1|P^{(1)}), \mathbb{Q}_{1|P^{(2)}}(1|P^{(2)}), \dots \mathbb{Q}_{1|P^{(l)}}(1|P^{(l)})\right]$, the output from $\mathcal{S}$, to obtain the global classification estimate $\mathbb{P}_{1|I}(1|I)$. Given a label y^* and a global prediction $\mathbb{P}_{1|I}(1|I)$ we use the *binary cross entropy loss* to determine the loss of the model, namely

$$L(I, y^*) = -y^* \log(\mathbb{P}_{1|I}(1|I)) - (1 - y^*) \log(\mathbb{P}_{0|I}(0|I)). \tag{8.1}$$

It is important to note that the weights for the CNN and the linear layers used to produce $\mathbb{Q}_{1|P^{(j)}}$ are shared across all patches $P^{(j)}$. Thus, the local network $\mathcal{S}$ used to produce $\mathbb{Q}_{1|P^{(j)}}$ must be able to identify features for class 1 and class 0 on a local scale. Furthermore, by performing an average instead of using a linear layer to obtain the output $\mathbb{P}_{1|I}(1|I)$, we are inherently forcing $\mathcal{S}$ to independently identify as many features of class 1 and class 0 as possible without providing global context, thereby enhancing interpretability of the learned features.

8.3. **Convergence Behavior of PatchNet.** The main limitation of such an approach is that the patch size parameters m', n' must be tuned based on the scale of features present in the data set. Intuitively, selecting too small values for m', n' forces $\mathcal{S}$ to learn just the area of structures in images as features, while selecting too large values of m', n' results in uninterpretable feature interactions. For many applications, domain expertise can be used for choosing an appropriate patch size.

Another limitation of this model is that when an image only contains few patches with features indicative of class 1, then the model would not be able to correctly classify it as class 1, since the patches P_1 with $\mathbb{Q}_{1|P_1}$ close to 1 would be out-voted by the patches P_0 with $\mathbb{Q}_{1|P_0}$ close to 0. This is illustrated via the following simple example: Suppose that our data space $\mathcal{D}$ consists only of two images: (1) The class 0 image that is simply a $m \times n \times 1$ image of 0 valued pixels; (2) the class 1 image that is a $m \times n \times 1$ image with all 0 valued pixels except for the pixels in the upper left $\frac{m}{2} \times \frac{n}{2} \times 1$ rectangle, which all have value 1 (see Figure 24). Suppose also that our model

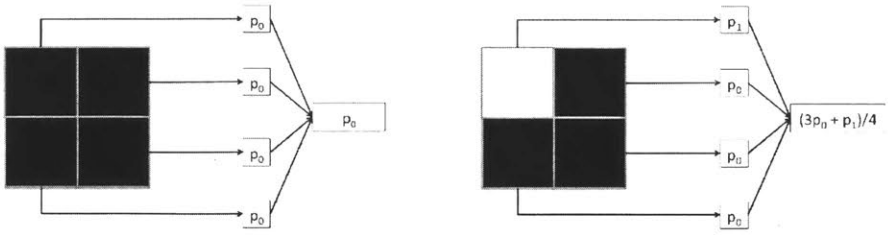


FIGURE 24. The class 0 images and class 1 images are depicted on the left and right respectively. As indicated, the images are segmented into 4 disjoint patches. We denote by p_0 the value of $\mathbb{Q}_{1|P_0}(1|P_0)$ for a patch P_0 containing all 0-valued pixels and by p_1 the value of $\mathbb{Q}_{1|P_1}(1|P_1)$ for a patch P_1 containing all 1-valued pixels.

uses $\frac{m}{2} \times \frac{n}{2} \times 1$ patches with a stride of $\frac{m}{2}$ in the first dimension and a stride of $\frac{n}{2}$ in the second dimension. That is, the model is trained on 4 patches for each class with only the upper left patch of class 1 images containing features indicative of class 1. Now suppose we train on a balanced set $\mathcal{T}$ of T images and labels $\{(I_k^{(1)}, y^*_{I_k^{(1)}}), (I_k^{(2)}, y^*_{I_k^{(2)}}), \ldots (I_k^{(T)}, y^*_{I_k^{(T)}})\}$ from both classes $k \in \{0, 1\}$, then we can deduce the values of $p_0 := \mathbb{Q}_{1|P_0}(1|P_0)$ and $p_1 := \mathbb{Q}_{1|P_1}(1|P_1)$ for patches P_0 containing all 0-valued pixels and patches P_1 containing all 1-valued pixels in closed form. Given $L(I, y^*)$ as defined in equation (1), the loss for all images in our training set is

$$L(\mathcal{T}) = \sum_{i=1}^{T} L(I_0^i, y^*_{I_0^i}) + \sum_{i=1}^{T} L(I_1^i, y^*_{I_1^i}) = -T\log(1-p_0) - T\log\left(\frac{3p_0+p_1}{4}\right).$$

To minimize this loss, we first analyze the derivative with respect to p_1, namely

$$\frac{\partial L}{\partial p_1} = -\frac{T}{3p_0+p_1},$$

which is always negative (since p_0, p_1, T ¿ 0) and hence closest to 0 when $p_1 = 1$. Taking derivatives with respect to p_0 yields

$$\frac{\partial L'}{\partial p_0} = -T\frac{3p_0+p_1+3p_0-3}{p_0(3p_0+p_1)},$$

which is 0 when $p_0 = \frac{1}{3}$. Substituting these values back into the global predictions, we obtain that $\mathbb{P}_{1|I_1}(1|I_1) = \frac{1}{2}$ for class 1 images I_1 and $\mathbb{P}_{1|I_0}(1|I_0) = \frac{1}{3}$ for class 0 images I_0. Hence, with a rounding threshold for class 1 of $.5 + \epsilon$, at convergence, our model would predict that all images belong to class 0. However, even though the resulting accuracy would only be 50%, the feature heatmap (constructed as described in Section 8.4) would still indicate that there is a feature representative of class 1 in the upper left corner of the image, since $\mathbb{Q}_{1|P^{(j)}}(1|P^{(j)}) = 1$ for all patches $P^{(j)}$ containing only 1-valued pixels.

This limitation explains mathematically why there is a trade-off between generalization error and interpretability: if there are few patches with features indicative of class 1 in class 1 images, then these patches must all output values $\mathbb{Q}_P$ close to 1 in order for $\mathbb{P}_{1|I_1}(1|I_1)$ to be closer to 1, and so the heatmap visualization will identify the regions of the image that are indicative of class 1. However, these patches can be outvoted by patches with features indicative of class 0 leading to a global misclassification of the image. See Appendix B for a generalized analysis of the convergence behavior and examples of the architecture displaying this behavior at convergence.

8.4. **Extracting Visualizations.** We now present how we can introspect layers of our model to easily provide visualizations of the features learned by the model. We refer to the gray-scale visualizations we generate as "heatmap" visualizations, since brighter pixels in the visualizations indicate that the model found a feature relevant to class 1, while darker pixels indicate a feature relevant to class 0. We provide two sets of heatmap visualizations and their corresponding pixel intensity histograms as follows.

The *global heatmap visualization* for an image is constructed by first computing $\mathbb{Q}_{1|P^{(i,j)}}(1|P^{(i,j)})$ for all patches $P^{(i,j)}$, where $P^{(i,j)}$ is the patch centered at location (i,j) in the first two dimensions of the original image (zero-padding is used for border locations), and then viewing an $m \times n$ image where each pixel at location (i',j') of the image is the value of $\mathbb{Q}_{1|P^{(i',j')}}(1|P^{(i',j')})$.

The *filter heatmap visualizations* are similar to global heatmap visualizations, except that a heatmap image is constructed for each filter in the last convolutional layer by using the output of the Tanh layer across all patches. We show the construction for filter 1 and apply the same construction for the other 63 filters. For each patch $P^{(i,j)}$ centered at location (i,j) in the first two dimensions of the original image (again with zero-padding for border locations), we compute the output of the Tanh layer and label the value corresponding to filter 1 as $F_1^{(i,j)}$ where $F_1^{(i,j)} \in [-1,1]$. Now we simply view the $m \times n$ image where each pixel at location (i',j') of the image is the min-max rescaled value of $F_1^{(i',j')}$ to get the filter heatmap for filter 1.

In this way, we generate and view 65 heatmap visualizations for each image fed to the model: 1 for the global heatmap and 64 for the filter heatmaps. As an aside, another approach to constructing filter or global heatmap visualizations is to average the predicted heatmap pixel values across all patches containing the given pixel. We found that this approach provided empirically inferior results in heatmap smoothness than the above described visualizations.

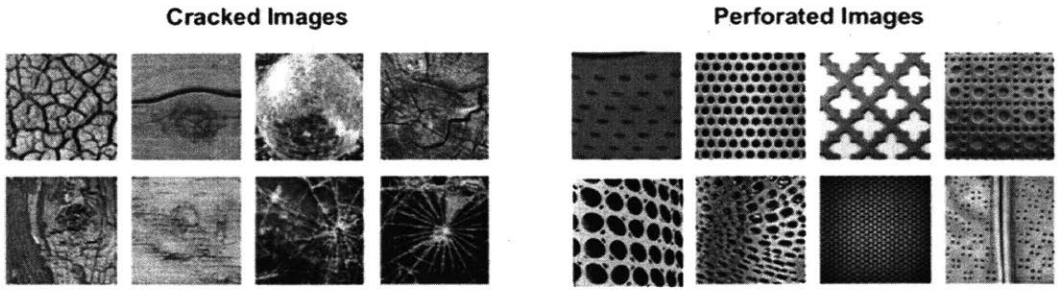


FIGURE 25. Samples of class 0 (cracked) images and class 1 (perforated) images are depicted on the left and right respectively.

8.5. **Precision-PatchNet.** In various applications, it is crucial to obtain high precision results, i.e., with few false positives. For example, a current problem with mammography for breast cancer detection is its high false positive rate: recent studies revealed that the chance of having a false positive result after 10 yearly mammograms is about 50-60% [22, 33, 35]. Such results can cause unnecessary worries and surgeries. We thus present a variation of PatchNet that is optimized for image classification tasks where having a low false positive rate matters more than overall accuracy.

For such *precision-based tasks*, the CNN $\mathcal{S}$ should additionally minimize the number of patches P outputting high values of $\mathbb{Q}_P$ for class 0 images, i.e., $\mathcal{S}$ should reduce the false positive rate for the patch-based classification in addition to being accurate. This can be achieved by a minor adjustment to the network, namely by introducing separate loss metrics for images in class 0 and class 1. For class 1 images, we keep the originally described binary cross entropy loss metric and hence the resulting loss function is as described in equation (1). However, for class 0 images, we introduce a binary cross entropy loss on $\mathbb{Q}_{1|P}(1|P)$ for each patch P and average it over all patches, i.e.,

$$\tilde{L}(I_0, 0) = \frac{1}{l} \sum_{i=1}^{l} \log(1 - \mathbb{Q}_{1|P^{(j)}}(1|P^{(j)})) \tag{8.2}$$

for an image I_0 with l patches. This loss function incentivizes the model to assign low probabilities $\mathbb{Q}_{1|P}(1|P)$ to each patch P in class 0, enforcing a low false positive rate among patches in class 0.

9. EXPERIMENTAL RESULTS

We now show the performance of PatchNet in providing interpretable features and analyze the trade-off between visual feature interpretability and generalization error for different patch sizes in applications to the classification of textures and melanoma. For each experiment we provide the

patch size, and training, validation, and test sets used to train our model. All of our models use the Adam optimizer with a learning rate of $5 \cdot 10^{-5}$, with batch sizes of either 8 or 16. We used Kaiming normal initialization [30] for all convolution layers and used min-max scaling to normalize the input images to have pixel values in the range $[0, 1]$. For all experiments, we augmented the training set using random horizontal and vertical reflections of the input images. Finally, to select the best model, we used a patience strategy [27], where we declared convergence when the model had not seen any improvement in validation loss for 2000 epochs. The software and hardware used for these applications is described in Appendix E and F.

9.1. **Describable Textures Dataset (DTD).** DTD [13] is a collection of real-world texture images annotated with "human-centric" attributes. The dataset consists of 47 classes of textures with 120 images per class. The image sizes range from $300 \times 300 \times 3$ to $640 \times 640 \times 3$. Although other texture datasets exist, such as CUReT [18], UMD [68] or UIUC [38], this dataset is the most extensive in terms of number of images per class.

We trained our model on two classes of images: (1) class 0 images are drawn from the "cracked" textures; (2) class 1 images are drawn from the "perforated" textures. We chose these classes as they have textures that are present on a local scale and these local textures generally repeat throughout the image. Samples from each of these classes are shown in Figure 25.

Since the images are of different sizes, we sampled random $128 \times 128 \times 3$ crops from the training set for each batch and used centered $128 \times 128 \times 3$ crops from the validation and test sets to validate and test. We used patches of size $15 \times 15 \times 3$ and a stride of 3 in each of the first two dimensions. As provided by DTD, we split the data into 40 images per class for each of the training, validation, and test sets. The classification performance is summarized in Figure 26 along with a comparison to the performance of SVM trained with the widely used *Parameter Free Threshold Adjacency Statistics* (PFTAS) features [15]. As shown in Appendix C, adding any other standard texture features caused the model to greatly overfit. PatchNet slightly under-performs the SVM model on test accuracy, even though both models over-fit the data.

Model	Train	Validation	Test
Neural Model	88.8%	76.3%	67.5%
Linear Model	99.3%	80.7%	73.3%

Reference\Prediction	Cracked	Perforated
Cracked	34	6
Perforated	20	20

FIGURE 26. The accuracy of our models and the confusion matrix for the DTD test set.

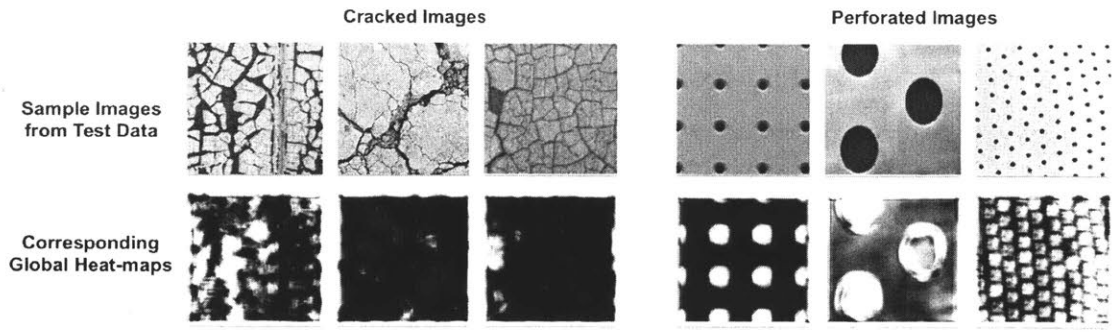


FIGURE 27. The global heatmaps for images sampled from the DTD test set.

The confusion matrix in Figure 26 shows that misclassification predominantly occurred for the perforated images. The global heatmaps for these images in Figure 27 show that the model correctly classified every perforation, but because in many images there is a larger amount of background than perforation, the model misclassified some perforated images. These results are thus in accordance with the mathematical derivation in Section 8.3.

As is also apparent from Figure 27, the model learned to identify dark contiguous regions in an image as perforations, while dark lines in an image were identified as cracks. Since the global heatmaps can be explained without any technical background, we would argue that the features learned by PatchNet are easier to interpret than the standardly used PFTAS texture features.

9.2. **ISBI-ISIC 2016 Melanoma Challenge.** Melanoma is a skin cancer that develops from pigmented lesions on the skin. The International Skin Imaging Collaboration (ISIC) Archive provided a collection of 1250 melanoma images (900 for training, 350 for testing) for the 2016 Melanoma Classification Challenge [28]. We used the 2016 melanoma detection challenge instead of the 2017 challenge, since testing data is currently only available for the 2016 challenge.

The data is split into benign (class 0) and malignant (class 1) skin lesions. Due to the high class imbalance with around 5 times more benign than malignant examples, we up-sampled the malignant class during training. The size of the original images is around $700 \times 1000 \times 3$. Due to

Model	Train	Validation	Test
Patch size 15	90.6%	86.1%	72.3%
Patch size 31	94.9%	88.3%	76.8%

Reference\Prediction	Benign	Malignant
Benign	393\136\256	16\13\48
Malignant	61\12\57	345\19\18

Reference\Prediction	Benign	Malignant
Benign	393\137\261	9\12\43
Malignant	32\9\45	381\22\30

FIGURE 28. From left to right: train, validation, and test accuracies for the PatchNet model trained using $15 \times 15 \times 3$ patches and $31 \times 31 \times 3$ patches; confusion matrix for the PatchNet model trained using $15 \times 15 \times 3$ patches; confusion matrix for the PatchNet model trained using $31 \times 31 \times 3$ patches.

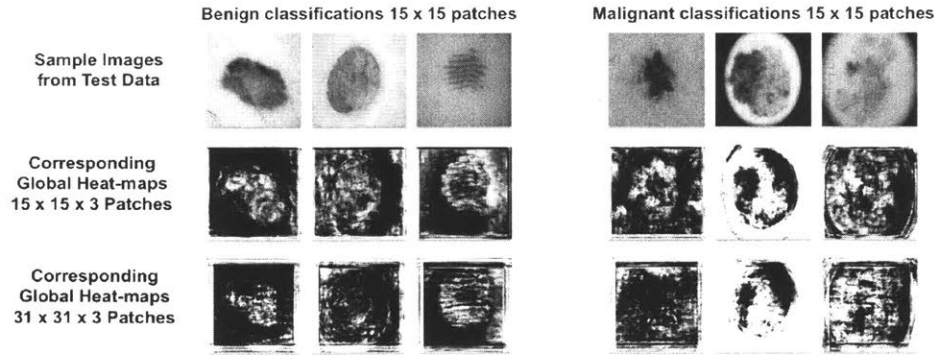


FIGURE 29. The global heatmaps for images sampled from the test set for patches of size $15 \times 15 \times 3$ and $31 \times 31 \times 3$.

memory limitations, we rescaled these images to $128 \times 170 \times 3$ prior to training, validating, and testing.

We now present the results of training two precision-based models. The first uses patches of size $15 \times 15 \times 3$ with a stride of 5 in each of the first two dimensions. The second uses patches of size $31 \times 31 \times 3$ with a stride of 10 in each of the first two dimensions. The resulting accuracies and confusion matrices are shown in Figure 28. Although both models over-fit, they both succeeded in minimizing the number of false positives during training and validation. In addition, as predicted by the mathematical derivation in Section 8.3, the larger patch size achieved a higher accuracy overall.

In Figure 29 we visualize the global heatmaps for sample test data using both models. As predicted in Section 8.3, there is a trade-off between generalization and visual interpretability for different patch sizes: the heatmaps for the model with larger patch size are more noisy as a consequence of training on patches with little overlap, but this model generalizes better. The heatmaps show that the model has learned to identify dark regions or spotted regions (known as globules) and irregular surroundings of regions (known as streaks). Interestingly, these are some of the features used by domain experts to classify melanoma. The 2016 ISBI-ISIC Melanoma Challenge also gives examples of images where these features were annotated. A comparison to the features found by PatchNet is shown in Figure 30.

As a side remark, the 2016 ISBI-ISIC Melanoma Challenge also illustrates the risks of using CNNs for medical applications. Since various malignant training examples had a lense around the melanoma, the model learned this as a feature. The second malignant example in Figure 29

shows that the model learned to classify the lens as part of a malignant lesion, when clearly the background is not part of the region of interest.

The ISIC also provided ground truth segmentations for the skin lesions. Interestingly, while most models for segmentation need to be trained on labeled segmentation data, by viewing the filter heatmaps from our models trained solely for classification, we were able to identify filters that provide lesion segmentations directly; see Figure 30.

Inspired by our segmentation results for melanoma, we trained PatchNet to also classify between images of human cell nuclei[1], an important preprocessing task in the classification of cancerous cells. In Figure 31 we present sample segmentation results obtained by PatchNet; for a careful analysis see Appendix D. Remarkably, although PatchNet was not trained on any labeled data for nucleus segmentation, it achieves precise segmentations regardless of the proximity of multiple nuclei and *halo* (i.e., intensity drop-off) effects.

10. Conclusion and Future Work

By forcing independent local classification decisions on patches, PatchNet is able to re-construct interesting global and filter heatmaps that provide easy introspection into learned features. In addition to interpretability, an interesting side effect of PatchNet is that select filters can directly be used for image segmentation. In future work, it will be interesting to further investigate the performance of our model for image segmentation. Since the described implementation is limited by existing hardware, an interesting line of future work is to investigate patch-level data parallelism techniques to increase our model's scalability.

[1]We would like to thank G. V. Shivashankar (National University of Singapore) for providing the images.

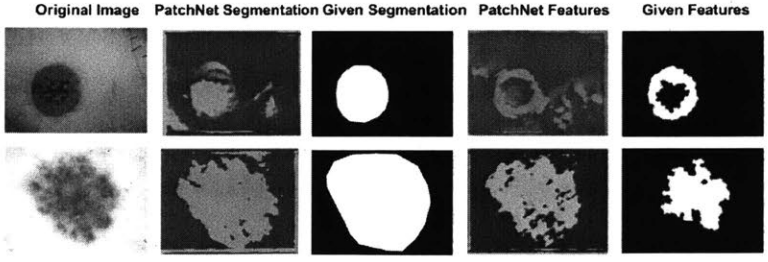


Figure 30. From left to right: original images sampled from the ISBI-ISIC melanoma dataset, segmentation provided by a filter from PatchNet, segmentation provided by ISBI-ISIC, features extracted by a filter from PatchNet, annotation of globules provided by ISBI-ISIC.

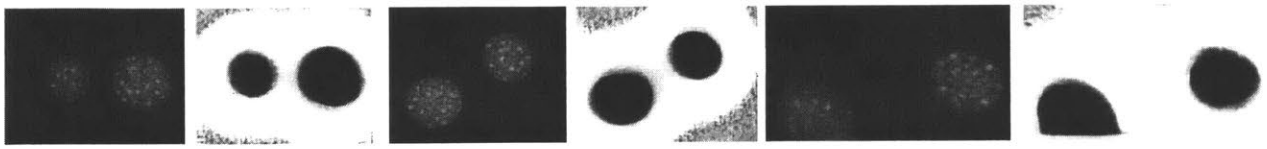


FIGURE 31. Three samples of nucleus segmentation as achieved by PatchNet for connective tissue cells in mice. The original image is shown on the left with the corresponding segmentation on the right.

ACKNOWLEDGEMENTS

I wish to thank Brendan McKay for some helpful advice in the use of the programs `nauty` and `Traces` [43]. I wish to thank Ali Soylemezoglu for his help in developing linear models to compare with our PatchNet model. I wish to thank Charles Durham for his help and guidance in developing the PatchNet architecture, the visualization techniques for PatchNet, and the hardware used to train PatchNet. I wish to thank Liam Solus for all his help and guidance in the entirety of our analysis of Markov Equivalence Classes. Most importantly, I wish to thank Caroline Uhler for her immense support and guidance over the past year and a half in our Markov Equivalence analysis, PatchNet development, and for helping me develop as a researcher.

APPENDIX A. BASIC DEFINITIONS AND NOTATION

A.1. **Graphs.**

A.1.1. *Basic definitions.* An *(undirected) graph* G is a pair of sets (V, E) in which V is some *node* set and E is the *edge* set, where an *edge* is taken to be an unordered pair of nodes $\{i, j\}$ for $i, j \in V$. We say that two nodes i, j are *adjacent* in $G = (V, E)$ if $\{i, j\} \in E$, or equivalently, we say that i is a *neighbor* of j and vice versa. The *neighborhood* of a node $i \in V$ in a graph $G = (V, E)$ is the set of all neighbors of i in G including the node i itself. A node is said to be *incident* to an edge $e \in E$ if i is one of the two defining nodes of e. The *degree* of a node i in a graph G is the number of edges incident to i in G, and it is denoted by $\deg_G(i)$. When the graph G is understood, we often write $\deg(i)$. A node i is called a *leaf* of G when $\deg(i) = 1$. A *path* in an undirected graph G is an alternating sequence of nodes and edges $(v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1})$ in which $e_i = \{v_i, v_{i+1}\}$ for all $i \in [n]$. A *cycle* in an undirected graph G is an alternating sequence of nodes and edges $(v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1})$ in which $e_i = \{v_i, v_{i+1}\}$ for all $i \in [n]$, and $v_1 = v_{n+1}$. A graph $G = (V, E)$ is called *connected* if there exists a path in G between every pair of nodes in V. An undirected graph is called a *forest* if it contains no cycles. A *tree* is a connected forest.

A graph $\mathcal{G} = (V, E)$ is *directed* if the edge set E is a set of *arrows*, where an *arrow* is defined to be any ordered pair of nodes, denoted (i, j) or $i \to j$ for $i, j \in V$. In an arrow $i \to j$ the node i is referred to as the *tail* of the arrow and the node j is referred to as the *head*. A *directed cycle* in a directed graph $\mathcal{G}$ is an alternating sequence of nodes and arrows $(v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1})$ in which $e_i = v_i \to v_{i+1}$ for all $i \in [n]$, and $v_1 = v_{n+1}$. A *directed acyclic graph* (or DAG) is a directed graph that does not contain any directed cycles. A node i is called a *parent* of node j in a directed graph $\mathcal{G}$ if $i \to j \in E$. In this case, the node j is called a *child* of node i. A node j is called a *descendant* of a node i if there is a directed path in $\mathcal{G}$ from i to j. In this case, then node i is an *ancestor* of j. We denote the set of parents, descendants, and ancestors of node i, respectively, by $\mathrm{pa}(i)$, $\mathrm{de}(i)$, and $\mathrm{an}(i)$. The set of *non-descendants* of i is $\mathrm{nd}(i) := V \backslash (\{i\} \cup \mathrm{de}(i))$.

A *partially directed graph* is a pair $G = (V, E)$ with node set V for which the set E consists of some edges $\{i, j\}$ and some $u \to v$ for $i, j, u, v \in V$. Both, DAGs and partially directed graphs have an underlying *skeleton*, which is the undirected graph given by replacing every arrow $i \to j$ with the edge $\{i, j\}$. The collection of all partially directed graphs include both, directed and undirected graphs, as subcollections. For this reason, we make the following definitions in the

setting of partially directed graphs, and simply note that they specify to directed and undirected graphs accordingly. A *subgraph* of a partially directed graph $C = (V, E)$ is a partially directed graph $C' = (V', E')$ for which $V' \subset V$ and $E' \subset E$. An *induced subgraph* of C is a subgraph $C' = (V', E')$ of C for which an edge or arrow $e \in E$ is in E' if and only if its two incident nodes are in V'. We often call C' the subgraph *induced* by V' and denote this by writing $C' = \langle V' \rangle$. We say that two partially directed graphs $C = (V, E)$ and $C' = (V', E')$ are *isomorphic* if there is a bijection $\varphi : V \longrightarrow V'$ such that $\{i, j\}$ is an edges of C if and only if $\{\varphi(i), \varphi(j)\}$ is an edge of C' and $i \to j$ is an arrow of C if and only if $\varphi(i) \to \varphi(j)$ is an arrow of C'. A *path* in a partially directed graph G is an alternating sequence of nodes and edges/arrows $(v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1})$ in which $e_i = \{v_i, v_{i+1}\}$ (or $e_i = v_i \to v_{i+1}$) for all $i \in [n]$. If the path includes no arrows it is called *undirected*, and if includes at least one arrow it is called *directed*. A *cycle* in an undirected graph G is an alternating sequence of nodes and edges/arrows $(v_1, e_1, v_2, e_2, \ldots, e_n, v_{n+1})$ in which $e_i = \{v_i, v_{i+1}\}$ (or $e_i = v_i \to v_{i+1}$) for all $i \in [n]$, and $v_1 = v_{n+1}$. If the cycle contains no arrows it is called *undirected* and if it includes at least one arrow it is called *directed*. A *chain graph* is a partially directed graph containing no directed cycles. The *chain components* of a chain graph are the connected components of its subgraph consisting of all its edges.

A.1.2. *Some important graph classes and their notation.* We now define some special classes of graphs that we will consider in this paper, and specify their notation. The *p-path* is the undirected graph $I_p := ([p], E)$ for which $E := \{\{i, i+1\} : i \in [n-1]\}$. The *p-cycle* is the undirected graph $C_p := ([p], E)$ for which $E := \{\{i, i+1\} : i \in [n-1]\} \cup \{\{1, n\}\}$. A *chordless cycle* in a graph G is any induced subgraph of G that is a cycle. A graph is called *chordal* if every chordless cycle in G is a 3-cycle. The *complete graph* on p nodes is the undirected graph $K_p := ([p], E)$ for which $E := \{\{i, j\} : i, j \in [p]\}$. Given two disjoint node sets V and W for which $\#V = p$ and $\#W = q$, the *complete bipartite graph* on V and W is the undirected graph $K_{p,q} := \{\{i, j\} : i \in V, j \in W\}$. The *p-star* is the complete bipartite graph $K_{1,p}$, and we often denote the p-star by $G_1(p)$. For $p \neq 1$, the *center node* of $G_1(p)$ is the unique node of degree p, and for $p = 1$ the center of $G_1(p)$ is chosen to be one of the two nodes of the graph. A *bistar* can be thought of as a gluing of two stars in which a leaf node of one star is glued to the the center node of the other star. We denote the bistar given by gluing a leaf of $G_1(q+1)$ to the center node of $G_1(p)$ by $G_2(p, q)$. Equivalently, the bistar $G_2(p, q)$ can be defined by attaching p leaves to one node of the 2-path I_2 and q leaves to the other node. More generally, we can define the graph $G_p(q_1, q_2, \ldots, q_p)$ to be the undirected graph given by attaching q_i leaves to node i of the p-path I_p. A *rooted* undirected graph $G = (V, E)$ is an undirected graph with a unique distinguished node $r \in V$ called the *root* of G. A *complete binary tree* is a rooted tree in which the root node has degree two and every other non-leaf node has degree three. A *circulant* on p nodes is a graph whose nodes are identified with $\mathbb{Z}/p\mathbb{Z}$, and whose edges are given by a specified *connection set* $C \subset \mathbb{Z}/p\mathbb{Z}$. In the undirected setting, we assume C is closed under additive inverses. The *circulant graph* on p nodes with connection set C is denoted $X(p, C)$ and has edges $\{i, j\}$ for all pairs i and j satisfying $i - j \in C$. We often abbreviate the connection C via a subset of $\left[\frac{p}{2}\right]$ by omitting the additive inverse of each element.

A.1.3. *Independent sets and vertex covers.* In this subsubsection we let $G = (V, E)$ be a undirected graph. An *independent set* in G is subset of mutually non-adjacent nodes $S \subset V$. A *vertex cover* of G is a subset of nodes $S \subset V$ for which every edges of G is incident to some node in S. The complement in V of an independent set is always a vertex cover in V. In particular, the complement of a maximum independent set is a minimum vertex cover. The *independence number of* G is the size of any maximum independent subset of G, and the *Fibonacci number of* G is the total number of independent sets of G. Let $\alpha_k(G)$ denote the number of independent subsets of size k in G, $\alpha(G)$ denote the independence number of G and $F(G)$ denote the Fibonacci number of G. These

numbers collect into the polynomial generating function

$$I(G;x) := \sum_{k\geq 0} \alpha_k(G)x^k,$$

which we call the *independence polynomial of* G. In particular, $I(G;1) = F(G)$.

A.2. **DAG Models.** Let $(X_v : v \in V)$ denote a vector of random variables that are indexed by the nodes of some directed acyclic graph $D = (V, E)$. The set of arrows and non-arrows in D encode a set of conditional independence constraints on random variables X_v, which are called the Markov properties of the DAG D. The *directed acyclic graphical model* associated to D is the family of multivariate probability distributions that abide by the Markov properties of D. To specify the Markov properties we will use, we first recall the a notion of directed separation (or equivalently, directed connectedness) for directed graphs.

An *undirected path* in the DAG D is an alternating sequence of nodes and arrows

$$\pi = (v_1, e_1, v_2, e_2, \ldots, e_{n-1}, v_n)$$

in which the arrow e_i is either $v_i \to v_{i+1}$ or $v_i \leftarrow v_{i+1}$ for all i. A *collider* on an undirected path in D is any subsequence $(v_1, e_1, v_2, e_2, v_3)$ such that $e_1 = v_1 \to v_2$ and $e_2 = v_2 \leftarrow v_3$. Given a DAG D, we consider the graphical model associated to D by the *directed global Markov property.* Two nodes u and v in D are said to be *d-connected* given $C \subset V\backslash\{u, v\}$ if there exists an undirected path π from u to v for which

(1) all colliders on π are in $C \cup \text{an}(C)$, and
(2) there is no non-collider on π that is also in C.

If A, B, and C are disjoint subsets of V with A and B being nonempty, then C is said to *d-separate* A and B if there are no $u \in A$ and $v \in B$ that are d-connected given C. The *directed global Markov property* associates the DAG $D = (V, E)$ with the set of conditional independence relations $X_A \perp\!\!\!\perp X_B \mid X_C$ for all triples A, B, and C for which C d-separates A and B. Oftentimes, multiple DAGs have the same set of d-separation relations, and thereby encode the same set of conditional independence statements. If two DAGs encode the exact same conditional independence relations with respect to the directed global Markov property we call them *Markov equivalent.* Given a DAG $D = (V, E)$, the collection of all DAGs that are Markov equivalent to D, including D itself, is called the *Markov equivalence class* of D. In this paper we denote the Markov equivalence class of D by $[D]$. There is a well-known characterization of when two DAGs are Markov equivalent, and it is given in terms of their skeleta and their set of immoralities. An *immorality* in a DAG D is a triple of node (a, b, c) for which D contains the arrows $a \to b$ and $b \leftarrow c$ but D does not contain either of the arrows $a \to c$ or $a \leftarrow c$.

Theorem 35. [64, Verma and Pearl] *Two DAGs are Markov equivalent if and only if they have the same skeleton and the same set of immoralities.*

For a more thorough treatment on the basics of graphical models in algebra and combinatorics see [19, Chapter 3].

A.3. **Essential graphs.** Let $\mathcal{G}$ be a DAG and let $[\mathcal{G}]$ denote the MEC containing $\mathcal{G}$. An *essential arrow* in $\mathcal{G}$ is an arrow which appears in all elements of $[\mathcal{G}]$. That is, $i \to j$ is an essential arrow of $\mathcal{G}$ if and only if $i \to j$ is an arrow of $\mathcal{G}'$ for all $\mathcal{G}' \in [\mathcal{G}]$. We also refer to such an arrow as an essential arrow of the MEC $[\mathcal{G}]$. The *essential graph* of the class $[\mathcal{G}]$ is the partially directed graph whose set of arrows is precisely the essential arrows of $[\mathcal{G}]$ and whose set of edges is those edges $\{i, j\}$ in the skeleton of $\mathcal{G}$ that support non-essential edges in $\mathcal{G}$. We denote the essential graph of $[\mathcal{G}]$ by $\widehat{\mathcal{G}}$. Recall that the *chain components* of a chain graph are the connected components of its subgraph consisting of all its undirected edges. We call the connected directed components of an essential graph its *essential components.* An arrow $i \to j$ is called *strongly protected* in a chain

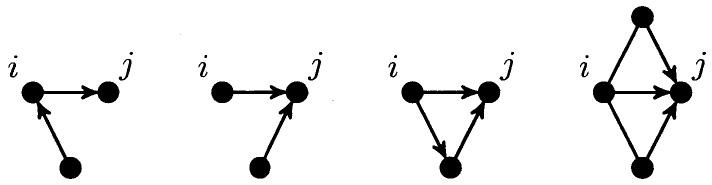


FIGURE 32. The edge $i \to j$ is strongly protected if it occurs in any of the above configurations.

graph if it occurs in any one of the configurations depicted in Figure 32. The following theorem provides a characterization of the partially directed graphs that are essential graphs of MECs.

Theorem 36. [2, Andersson et. al] *A partially directed graph $G = (V, E)$ is the essential graph $\widehat{\mathcal{G}}$ for some DAG $\mathcal{G}$ if and only if G satisfies the following conditions:*

(1) G is a chain graph.
(2) Every chain component of G is chordal.
(3) The configuration $i \to j - k$ does not occur as an induced subgraph of G.
(4) Every arrow $i \to j$ is strongly protected in G.

A detailed development of the theory of essential graphs is found in [2].

A.4. **Enumerative definitions.** A *composition* of a positive integer n into k parts is an ordered sum of k positive integers equaling n, i.e.

$$c_1 + c_2 + \cdots + c_k = n.$$

The positive integers $c_1, \ldots, c_k$ are called the *parts* of the composition. A partition of the positive integer n is a sequence of integers $\lambda := (\lambda_1, \lambda_2, \ldots)$ satisfying $\sum_i \lambda_i = n$ and $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$. The nonzero integers λ_i are called the *parts* of the partition, and we say that λ has k parts if $\#\{i : \lambda_i > 0\} = k$. We let $\mathbb{P}[k, n]$ denote the set of partitions of n with k parts, and we let $\mathbb{P}[j, k, n]$ denote the partitions of n with k parts with largest possible part j. Note that our notation differs slightly from the notation used in [59], in which $p(j, k, n)$ denotes the number of partitions of n with *at most* k parts and largest part being j. If the partition λ has m_i parts equaling i then we may write $\lambda = \langle 1^{m_1}, 2^{m_2}, \ldots \rangle$. When $\lambda \in \mathbb{P}[j, k, n]$, then $\lambda = \langle 1^{m_1}, 2^{m_2}, \ldots, j^{m_j} \rangle$ and so λ is uniquely identified with the integer vector $(m_1, m_2, \ldots, m_j)$. In this case, we simply write $(m_1, m_2, \ldots, m_j) \in \mathbb{P}[j, k, n]$. A wealth of enumerative combinatorics, including a very complete treatment of compositions and integer partitions, can be found in [59].

APPENDIX B. GENERALIZED CONVERGENCE BEHAVIOR OF PATCHNET

Generalizing from the results in Section 3.3, we analyze how the value of $\mathbb{Q}_P$ for each patch P is related to the fraction of features indicative of class 1 present in class 1 images, the fraction of features indicative of class 0 present in class 0 images, and the fraction of features shared among both classes. Intuitively, the model will assign low $\mathbb{Q}_P$ outputs for each patch P with a feature indicative solely of class 0, high $\mathbb{Q}_P$ outputs for each patch P with a feature indicative solely of class 1, and roughly $\mathbb{Q}_P = .5$ for patches with features shared between class 0 and 1. In this section, we perform a case analysis on simplified images that contain only features that are indicative of class 1 or features that are shared between class 0 and class 1. We mathematically determine the exact values of $\mathbb{Q}_P$ for each patch after convergence.

Due to the nature of our loss function, we can mathematically analyze the behavior of the model to understand the relationship between the visual heatmap and presence of class 1 features in the dataset under some simplifying assumptions. Namely, suppose that the images in the dataset contain only features, f_c, common to both classes of the image or features, f_1, indicative of class 1.

Now suppose that on average there are a patches in class 1 that are indicative of f_c and b patches in class 1 that are indicative of f_1, and that all $a+b$ patches of class 0 are indicative of f_c with $a > b$. Let us further assume that each patch contains only one of f_c or f_1, and so $\mathbb{Q}_{1|P^{(j)}}(1|P^{(j)}) = p_0$ for all j such that patch $P^{(j)}$ contains f_c and $\mathbb{Q}_{1|P^{(j)}}(1|P^{(j)}) = p_1$ for all j such that $P^{(j)}$ contains f_1. Then, as in Section 3.3, at convergence the model minimizes the loss

$$\begin{aligned} L &= -\log\left(1 - \frac{(a+b)p_0}{(a+b)}\right) - \log\left(\frac{ap_0 + bp_1}{a+b}\right) \\ &= -\log(1-p_0) - \log\left(\frac{ap_0 + bp_1}{a+b}\right) \end{aligned}$$

In order to minimize the loss, we set the derivatives with respect to p_0 and p_1 equal to 0. The derivative with respect to p_1 is:

$$\begin{aligned} \frac{\partial L}{\partial p_1} &= -\frac{a+b}{ap_0 + bp_1}\frac{b}{a+b} \\ &= -\frac{b}{ap_0 + bp_1} \end{aligned}$$

This derivative is always less than 0 as $a, b, p_1, p_0 \geq 0$. Hence, the model will simply try to maximize the value of p_1 to bring the derivative closer towards 0. Thus, at convergence, $p_1 = 1$. Now examining the derivative with respect to p_0:

$$\begin{aligned} \frac{\partial L}{\partial p_0} &= 0 \\ \implies -\frac{1}{p_0 - 1} - \frac{a+b}{ap_0 + bp_1}\frac{a}{a+b} &= 0 \\ \implies ap_0 + bp_1 + ap_0 - a &= 0 \\ \implies ap_0 + b + ap_0 - a &= 0 \\ \implies p_0 &= \frac{a-b}{2a} \end{aligned}$$

Hence the largest possible value for p_0 is $\frac{1}{2}$, which occurs when $b = 0$. Note that we performed this analysis for $a > b$. If $a \leq b$, as p_0 is constrained to be in the range $[0, 1]$, the value of p_0 will simply be 0 at convergence.

As a concrete example of this convergence behavior in practice, we can examine the behavior of the model on a sample binary data set where class 0 consists of all black 128×128 images (each pixel has value 0), and class 1 consists of 128×128 images with a white square of size 64×64 in the upper left hand corner (each pixel has value 1) (a larger version of the sample image in Section 3.3).

We train our model using 17×17 patches and a stride of 17 in either direction. In this case, there are some patches that contain both f_c and f_1. Yet as there are only a few such patches, the actual values of p_0 and p_1 should only be slightly noisy. We use zero padding and a stride size of 1 to visualize the outputs of $\mathbb{Q}$ on each of the $128 \cdot 128 = 16384$ patches centered at each pixel value in the original image. Since the f_1 features are contained in a 64×64 square in the upper left corner of the image, we claim that there are approximately $b = 64 \cdot 64 = 4096$ patches that contain f_1.

Now, by our calculations, we expect that $p_0 = \frac{16384 - 2 \cdot 4096}{2 \cdot (16384 - 4096)} = \frac{1}{3}$ and that $p_1 = 1$. That is, we expect our model to output a value of $\frac{1}{3}$ for patches containing only f_c and 1 for patches containing

Feature	Train	Validation	Test
PFTAS + Haralick	99.9%	83.1%	45.7%
PFTAS + Zernike + Haralick	100.0%	82.6%	44.7%
PFTAS	99.3%	80.7%	73.3%
PFTAS + Zernike Moments	99.8%	80.0%	46.8%
Zernike + Haralick	92.4%	78.8%	46.3%
Haralick	91.9%	78.4%	49.5%
Zernike	65.7%	64.3%	45.6%

TABLE 2. Training, Validation, and Test results for SVM across DTD [13] data splits.

Feature	Train	Validation	Test
PFTAS + Zernike + Haralick	98.9%	82.9%	46.2%
PFTAS + Haralick	98.9%	82.8%	44.5%
PFTAS + Zernike Moments	98.3%	81.6%	48.5%
PFTAS	98.0%	81.4%	65.0%
Zernike + Haralick	90.3%	79.2%	45.9%
Haralick	89.1%	77.6%	45.5%
Zernike	65.7%	64.6%	45.6%

TABLE 3. Training, Validation, and Test results for Logistic Regression across DTD [13] data splits.

Model	Train	Validation
Neural Model	94.9%	90.9%

TABLE 4. Training and validation results for PatchNet across the nucleus dataset.

only f_1. Indeed, after our model had converged to a local minimum, the model output a value of 0.333 for patches consisting of only f_c and output a value of 0.982 for a patch consisting of only f_1.

APPENDIX C. LINEAR MODELS FOR DTD

In this section, we describe all linear models used on the DTD data for cracked and perforated images [13]. The training, validation, and test accuracies for the SVM model and logistic regression model are shown in Table 2 and Table 3, respectively. We ran the linear models using popular texture features extracted using the Mahotas library [14]. From these tables it is apparent that all linear models overfit with these texture features. The best test accuracy was achieved by using the SVM model with only PFTAS features (as reported in Section 4.1). However, if we were to select our models based on validation accuracy, as is typically done in practice, the best model is SVM using both PFTAS and the Haralick features, which achieves a poor test accuracy of 45.7%, significantly worse than the test accuracy achieved by PatchNet.

Reference\Prediction	BJ	NIH3T3
BJ	428\68	47\14
NIH3T3	1\1	474\81

TABLE 5. Confusion matrix for training\validation data from the nucleus dataset.

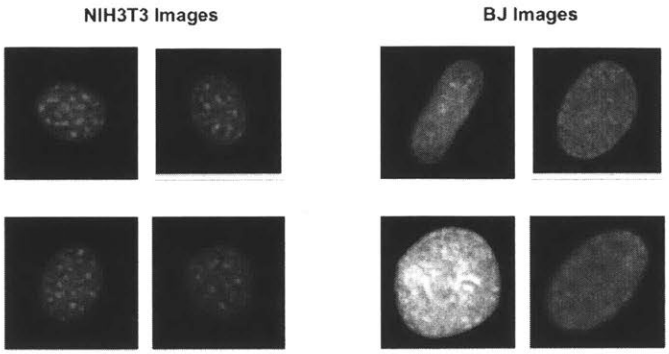


FIGURE 33. Samples of class 1 (NIH3T3) images and class 0 (BJ) images are depicted on the left and right respectively.

APPENDIX D. CELL NUCLEUS DATA

In this section, we discuss the results of running our model for the problem of extracting cell nuclei from images obtained by our collaborator G. V. Shivashankar (National University of Singapore). This task is an important preprocessing step for identifying cancerous cells in tissue images. The dataset of cell nuclei consists of connective tissue cells from humans (*BJ* cells) and mice (*NIH3T3* cells). Sample images of NIH3T3 nuclei and BJ nuclei are shown in Figure 33.

D.1. **Nucleus Crops.** We first run PatchNet on pre-segmented nuclei crops of BJ (class 0) and NIH3T3 (class 1). The images are of size $128 \times 128 \times 1$. We used 557 images of each class for training and 82 images of each class for validation. We trained using $17 \times 17 \times 1$ patches with a stride of 5. The resulting accuracies and confusion matrix given in Table 4 and 5 show that the model was able to achieve a high validation accuracy for this dataset.

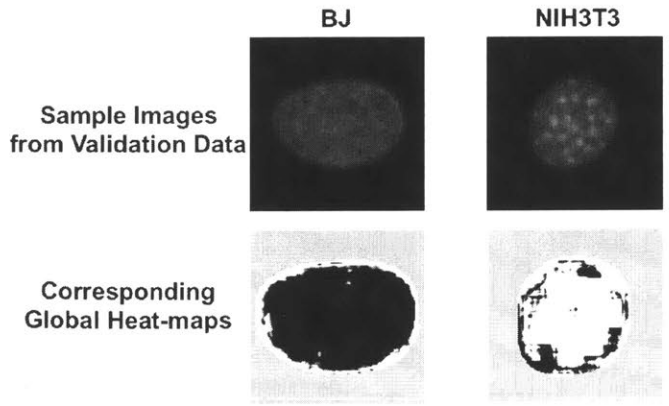


FIGURE 34. Samples of class 0 (BJ) images and class 1 (NIH3T3) images are depicted on the left and right respectively along with the corresponding global heatmaps pictured below.

We now present the global heatmaps in Figure 34. From the heatmaps, we see that the model very accurately learns the regions occupied by NIH3T3 and BJ nuclei. We also see that the model assigns a grey value to the background indicating that the background is common to both images. All 64 filter heatmaps are presented in Figure 35 and 36.

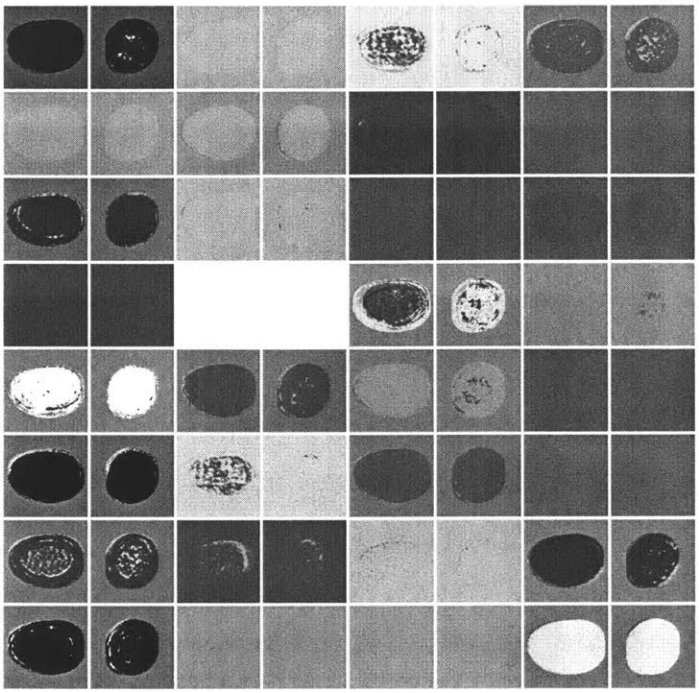


FIGURE 35. Four columns of 8 pairs of filter heatmaps with the class 0 filter heatmap appearing in the left of the column pairs and the class 1 filter heatmap appearing in the right of the column pairs.

As indicated by the large proportion of grey filter pairs, we see that not all filters are required to achieve high accuracy on this dataset, indicating that our model has more capacity than required for this problem. In addition, we see that some filters that are contributing to classification are able to pick up the bright dots on the NIH3T3 nuclei as features. These are known features indicative of mouse cells. We also note the presence of filters that seem similar to edge detectors on the internal regions of the nuclei.

D.2. **Nucleus Segmentation.** Inspired by our ability to find filters that segmented melanoma from the ISBI-ISIC challenge as presented in Section 4.2, we ran PatchNet on the original unsegmented $512 \times 512 \times 1$ images of BJ and NIH3T3 cells with a patch size of $17 \times 17 \times 1$ in order to determine if there were filters able to segment the nuclei without any preprocessing.

Representative examples of unsegmented nucleus images are shown in Figure 37. Note the following two interesting properties of these unsegmented images: (1) NIH3T3 images tend to have more nuclei per image than BJ images; (2) there are noticable "halo" effects, creating a region of higher pixel values around the nuclei, which makes the segmentation task difficult using standard segmentation methods.

Figure 38 and 39 show the results of two particular filters of PatchNet when trained on classifying between the unsegmented images. These filters accurately segment out the nuclei as dark ovals regardless of halo effects and regardless of the fact that 2 BJ nuclei appear together in an image. Figure 39 shows that filter 57 is able to segment out the cell nuclei and at the same time identifies

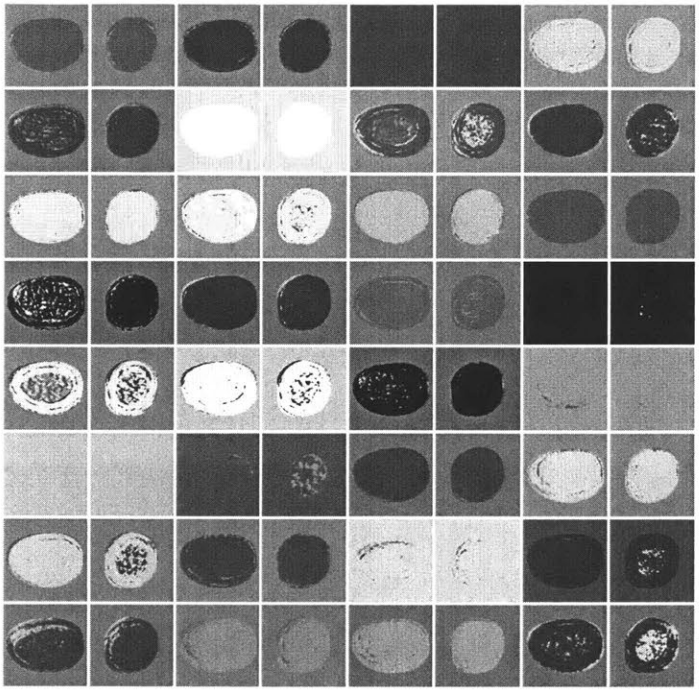


FIGURE 36. The remaining 4 columns of 8 pairs of filter heatmaps with the class 0 filter heatmap appearing in the left of the column pairs and the class 1 filter heatmap appearing in the right of the column pairs.

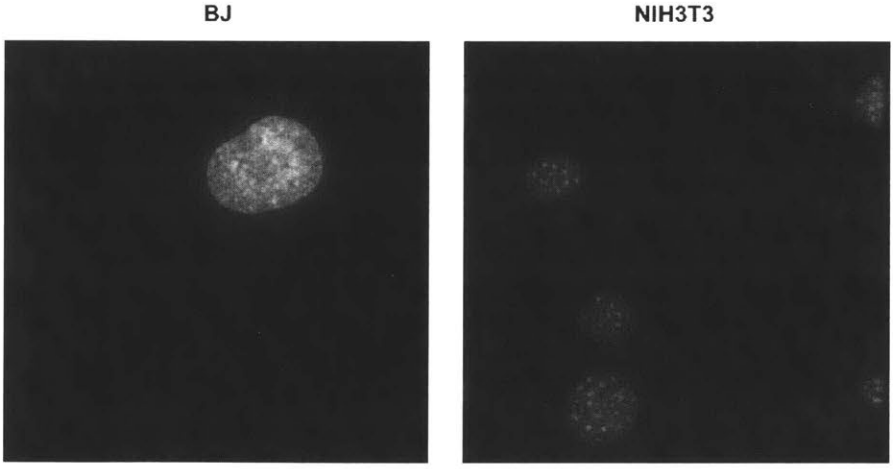


FIGURE 37. Samples of class 0 (BJ) images and class 1 (NIH3T3) images are depicted on the left and right respectively.

the texture features that identify NIH3T3 cells, namely the bright spots in the cell nuclei. These segmentation results are quite remarkable, taking into account that the model was not trained on labeled data for nucleus segmentation.

APPENDIX E. SOFTWARE

We ran our neural network models on PyTorch [50] 0.1.12 on Python 3.6 packaged under Anaconda [1] 4.3.17 with Nvidia driver version 375.51, Cuda version 8.0.61 and CuDNN version 5.1.0.

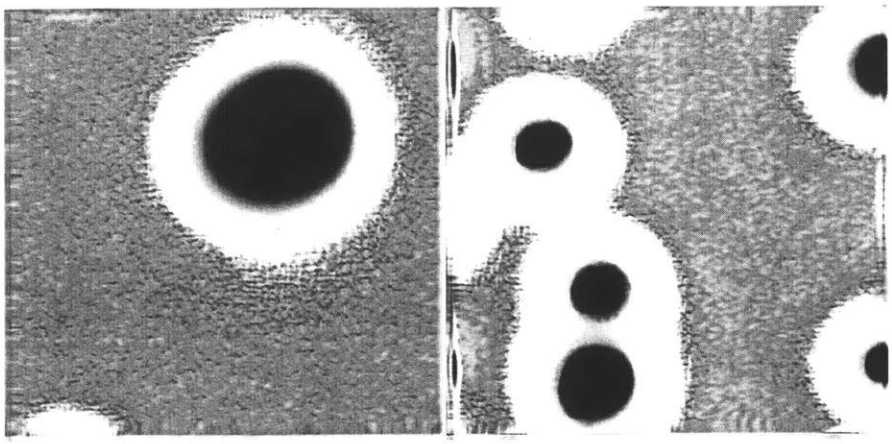


FIGURE 38. Heatmaps for filter 55 of PatchNet for the nuclei in Figure 37.

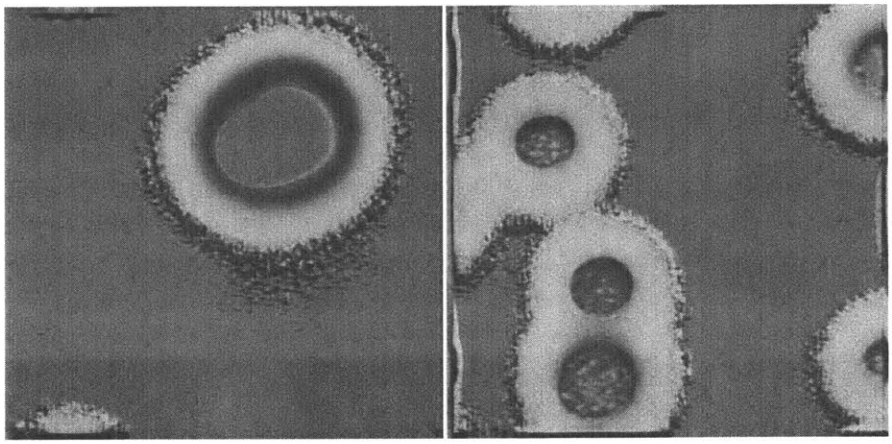


FIGURE 39. Heat maps for filter 57 of PatchNet for the nuclei in Figure 37.

We used SciKit-Learn [46], Numpy [67] and Mahotas [14] as tools while developing our neural and linear models. We used Facebook's Visdom [65] as a tool to visualize filters and display them for this paper.

APPENDIX F. HARDWARE

For training PatchNet, we used one server running Ubuntu 16.04.2 with an Intel i7-4930K at 3.40GHz with 56GB DDR3 RAM and two Nvidia Titan x (Pascal) with 12GB of GDDRX RAM.

REFERENCES

[1] Anaconda Software Distribution. Computer software. Vers. 2-2.4.0. Continuum Analytics, Nov. 2016. Web. `https://continuum.io`

[2] S. A. Andersson, D. Madigan, and M. D. Perlman. *A characterization of Markov equivalence classes for acyclic digraphs.* The Annals of Statistics 25.2 (1997): 505–541.

[3] V. C. Barbosa and J. L. Szwarcfiter. *Generating all the acyclic orientations of an undirected graph.* Information Processing Letters 72.1 (1999): 71–74.

[4] M. Bašić and A. Ilić. *On the clique number of integral circulant graphs.* Applied Mathematics Letters 22.9 (2009): 1406–1411.

[5] B. Bollobás. *The independence ratio of regular graphs.* Proceedings of the American Mathematical Society (1981): 433–436.

[6] B. Braun and L. Solus *Shellability, Ehrhart theory, and r-stable hypersimplices.* Submitted to Journal of Combinatorial Theory Series A. ArXiv preprint arXiv:1408.4713 (2015).

[7] J. Brown and R. Hoshino. *Independence polynomials of circulants with an application to music.* Discrete Mathematics 309.8 (2009): 2292–2304.

[8] J. Brown and R. Hoshino. *Well-covered circulant graphs.* Discrete Mathematics 311.4 (2011): 244–251.

[9] P. Cain. *Decomposition of complete graphs into stars.* Bulletin of the Australian Mathematical Society 10.01 (1974): 23–30.

[10] J. M. Carraher, D. Galvin, S. G. Hartke, A. J. Radcliff, and D. Stolee. *On the independence ratio of distance graphs.* ArXiv preprint arXiv:1401.7183 (2014).

[11] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, N. Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In Knowledge Discovery and Data Mining (KDD), 2015.

[12] D. M. Chickering. *Learning equivalence classes of Bayesian-network structures.* Journal of Machine Learning Research 2 (2002): 445–498.

[13] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In the Conference on Computer Vision and Pattern Recognition (*CVPR*), 2014.

[14] L.P. Coelho. *Mahotas: Open source software for scriptable computer vision.* Journal of Open Research Software. 1(1), p.e3. DOI: `http://doi.org/10.5334/jors.ac`

[15] L.P. Coelho et al. (2010) Structured Literature Image Finder: Extracting Information from Text and Images in Biomedical Literature. In: Blaschke C., Shatkay H. (eds) Linking Literature, Information, and Knowledge for Biology. Lecture Notes in Computer Science, vol 6004. Springer, Berlin, Heidelberg.

[16] E. Cohen and M. Tarsi. *NP-completeness of graph decomposition problems.* Journal of Complexity 7.2 (1991): 200–212.

[17] G. Cooper, C. Aliferis, R. Ambrosino, J. Aronis, B. Buchanan, R. Caruana, M. Fine, C. Glymour, G. Gordon, B. Hanusa, J. Janosky, C. Meek, T. Mitchell, T. Richardson, and P. Spirtes. An evaluation of machine-learning methods for predicting pneumonia mortality. Artificial Intelligence in Medicine, 9(2):107138, 1997.

[18] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink. *Reflectance and texture of real world surfaces.* ACM Transactions on Graphics, 18(1):134, 1999.

[19] M. Drton, B. Sturmfels, and S. Sullivant. *Lectures on Algebraic Statistics.* Vol. 39. Springer Science & Business Media, 2008.

[20] J. A. Ellis-Monaghan and C. Merino. *Graph polynomials and their applications I: The Tutte polynomial.* Structural Analysis of Complex Networks. Birkhäuser Boston, 2011. 219–255.

[21] J. A. Ellis-Monaghan and C. Merino. *Graph polynomials and their applications II: Interrelations and interpretations.* Structural Analysis of Complex Networks. Birkhäuser Boston, 2011. 257–292.

[22] J. G. Elmore, M. B. Barton, V. M. Moceri, S. Polk, P. J. Arena, and S. W. Fletcher. *Ten-year risk of false positive screening mammograms and clinical breast examinations.* The New England Journal of Medicine, 338: 1089-96, 1998.

[23] N. Friedman, M. Linial, I. Nachman and D. Peter. *Using Bayesian networks to analyze expression data.* Journal of Computational Biology 7 (2000): 601–620.

[24] M. R. Garey and D. S. Johnson. *Computers and intractability: a guide to the theory of NP-completeness.* A Series of Books in the Mathematical Sciences. WH Freeman and Company, New York, NY 25.27 (1979): 141.

[25] S. B. Gillispie. *Formulas for counting acyclic digraph Markov equivalence classes.* Journal of Statistical Planning and Inference 136.4 (2006): 1410-1432.

[26] S. B. Gillispie and M. D. Perlman. *Enumerating Markov equivalence classes of acyclic digraph models.* Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., 2001.

[27] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[28] D. Gutman, N. C. F. Codella, E. Celebi, B. Helba, M. Marchetti, N. Mishra, and A. Halpern. *Skin lesion analysis toward melanoma detection: A challenge at the International Symposium on Biomedical Imaging (ISBI) 2016,* hosted by the International Skin Imaging Collaboration (ISIC), 2016; arXiv:1605.01397.

[29] N. Hamada, H. Ikeda, S. Shiga-eda, K. Ushio, and S. Yamamoto. *On claw-decomposition of complete graphs and complete bigraphs.* Hiroshima Mathematical Journal 5.1 (1975): 33–42.

[30] K. He, X. Zhang, S. Ren, and J. Sun. *Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification.* In the International Conference on Computer Vision (*ICCV*), 2015.

[31] K. He, X. Zhang, S. Ren, and J. Sun. *Deep residual learning for image recognition.* In the Conference on Computer Vision and Pattern Recognition (*CVPR*), 2016.

[32] R. Hoshino. *Independence polynomials of circulant graphs.* Library and Archives Canada, 2008.

[33] R. A. Hubbard, K. Kerlikowske, C. I. Flowers, B. C. Yankaskas, W. Zhu, D. L. Miglioretti. *Cumulative probability of false-positive recall or biopsy recommendation after 10 years of screening mammography: a cohort study.* Annals of Internal Medicine, 155(8):481-92, 2011.

[34] J. D. Hunter. *Matplotlib: A 2D graphics environment.* Computing in Science and Engineering 9.3 (2007): 90–95.

[35] Independent UK Panel on Breast Cancer Screening. *The benefits and harms of breast cancer screening: an independent review.* Lancet. 380(9855):1778-86, 2012.

[36] R. M. Karp *Reducibility among combinatorial problems.* Complexity of Computer Computations. Springer US (1972): 85–103.

[37] D. P. Kingma and J. Ba. *Adam: A method for stochastic optimization*; The International Conference on Learning Representations (*ICLR*), 2015.

[38] S. Lazebnik, C. Schmid, and J. Ponce. *Sparse texture representation using local affine regions.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(8):21692178, 2005.

[39] T. Lei, R. Barzilay, and T. S. Jaakkola. *Rationalizing Neural Predictions.* Conference on Empirical Methods in Natural Language Processing, 2016.

[40] V. E. Levit and E. Mandrescu. *The independence polynomial of a graph – a survey.* Proceedings of the 1st International Conference on Algebraic Informatics. Vol. 233254. 2005.

[41] C. Lin, and T-W. Shyu. *A necessary and sufficient condition for the star decomposition of complete graphs.* Journal of Graph Theory 23.4 (1996): 361–364.

[42] J. H. van Lint and R. M. Wilson. *A Course in Combinatorics.* Cambridge University Press, 2001.

[43] B. D. McKay and A. Piperno. *Practical graph isomorphism, II.* Journal of Symbolic Computation 60 (2014): 94–112.

[44] C. Meek. *Causal inference and causal explanation with background knowledge.* Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence (1995): 403–410.

[45] J. Pearl. *Causality: Models, Reasoning, and Inference.* Cambridge University Press, Cambridge, 2000.

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, . Duchesnay. *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research (JMLR) 12, pp. 2825-2830, 2011.

[47] S. Poljak. *A note on stable sets and colorings of graphs.* Commentationes Mathematicae Universitatis Carolinae 15.2 (1974): 307–309.

[48] H. Prodinger and R. F. Tichy. *Fibonacci numbers of graphs.* Fibonacci Quarterly 20.1 (1982): 16–21.

[49] H. Prodinger and R. F. Tichy. *Fibonacci numbers of graphs. II.* Fibonacci Quarterly 21.3 (1983): 219–229.

[50] PyTorch. `http://pytorch.org`

[51] J. M. Robins, M. A. Hernán and B. Brumback. *Marginal structural models and causal inference in epidemiology.* Epidemiology 11.5 (2000): 550–560.

[52] R. W. Robinson. *Counting labeled acyclic digraphs.* New Directions in the Theory of Graphs. Academic Press (1977): 239–273.

[53] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. *ImageNet Large Scale Visual Recognition Challenge.* In the International Journal of Computer Vision (*IJCV*), 2015.

[54] K. Simonyan, A. Vedaldi, and A. Zisserman. *Deep inside convolutional networks: Visualising image classification models and saliency maps*, 2013; arXiv:1312.6034.

[55] K. Simonyan and A. Zisserman. *Very deep convolutional networks for large-scale image recognition.* In the International Conference on Learning Representations (*ICLR*), 2015.

[56] N. J. Sloane. *The On-Line Encyclopedia of Integer Sequences.* (2003).

[57] P. Spirtes, C. N. Glymour and R. Scheines. *Causation, Prediction, and Search.* MIT Press, Cambridge, 2001.

[58] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. *Striving for simplicity: The All Convolutional Net*, 2014; arXiv:1412.6806.

[59] R. Stanley. *Enumerative Combinatorics, vol. 1.* Wadsworth and Brooks/Cole, Pacific Grove, CA, 1986; second printing. (1996).

[60] B. Steinsky. *Enumeration of labelled chain graphs and labelled essential directed acyclic graphs.* Discrete Mathematics 270.1 (2003): 267-278.

[61] M. Tarsi. *Decomposition of complete multigraphs into stars.* Discrete Mathematics 26.3 (1979): 273–278.

[62] K. Ushio. *G-designs and related designs.* Discrete Mathematics 116.1 (1993): 299–311.

[63] K. Ushio, S. Tazawa and S. Yamamoto. *On claw-decomposition of a complete multipartite graph.* Hiroshima Mathematical Journal 8.1 (1978): 207–210.

[64] T. Verma and J. Pearl. *An algorithm for deciding if a set of observed independencies has a causal explanation.* Proceedings of the Eighth International Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann Publishers Inc., 1992.

[65] Visdom. `https://github.com/facebookresearch/visdom`

[66] S. Wagner. *Asymptotic enumeration of extensional acyclic digraphs.* Algorithmica 66.4 (2013): 829–847.

[67] S. van der Walt, S. Colbert and Gal Varoquaux. *The NumPy Array: A Structure for Efficient Numerical Computation.* Computing in Science & Engineering, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37

[68] Y. Xu, H. Ji, and C. Fermuller. *Viewpoint invariant texture description using fractal analysis.* In the International Journal of Computer Vision (*IJCV*), 83(1):85100, jun 2009.

[69] M. D. Zeiler and R. Fergus. (2013) *Visualizing and understanding convolutional networks.* In the European Conference on Computer Vision (*ECCV*) (1) 2014: 818-833.

[70] P. Zwiernik. *Semialgebraic Statistics and Latent Tree Models.* CRC Press, 2015.