Large-Scale Object Classification using Label Relation Graphs: Supplemental Material

Jia Deng[†*], Nan Ding[*], Yangqing Jia[*], Andrea Frome[*], Kevin Murphy[*], Samy Bengio[*], Yuan Li[*], Hartmut Neven[*], Hartwig Adam[*]

University of Michigan[†], Google Inc.[*]

1 Proofs

We provide all proofs in this section. The numbering of definitions, theorems and lemmas is the same as in the main paper.

1.1 Hierarchy and Exclusion (HEX) Graphs

Definition 1. *A HEX graph $G = (V, E_h, E_e)$ is a graph consisting of a set of nodes $V = \{v_1, \ldots, v_n\}$, directed edges $E_h \subseteq V \times V$, and undirected edges $E_e \subseteq V \times V$, such that the subgraph $G_h = (V, E_h)$ is a directed acyclic graph (DAG) and the subgraph $G_e = (V, E_e)$ has no self loop.*

Definition 2. *An assignment (state) $y \in \{0,1\}^n$ of labels V in a HEX graph $G = (V, E_h, E_e)$ is legal if for any $(y_i, y_j) = (1,1)$, $(v_i, v_j) \notin E_e$ and for any $(y_i, y_j) = (0,1)$, $(v_i, v_j) \notin E_h$. The state space $S_G \subseteq \{0,1\}^n$ of graph G is the set of all legal assignments of G.*

We now introduce some notations for further development. Let $\alpha(v_i)$ the set of all ancestors of $v_i \in V$ and $\bar{\alpha}(v_i) = \alpha(v_i) \cup v_i$ (ancestors and the node itself). Let $\sigma(v_i)$ be the set of all descendants of $v_i \in V$ and $\bar{\sigma}(v_i) = \sigma(v_i) \cup v_i$. Let $\epsilon(v_i)$ be the set of exclusive nodes, those sharing an exclusion edge with v_i. Let $o(v_i)$ be the set of overlapping nodes, those sharing no edges with v_i. Depending on the context we will add a subscript such as $\alpha_G(v)$ to emphasize that it is w.r.t. to graph G.

Consistency

Definition 3. *A HEX graph $G = (V, E_h, E_e)$ is consistent if for any label $v_i \in V$, there exists two legal assignments $y, y' \in \{0,1\}^n$ such that $y_i = 1$ and $y'_i = 0$.*

Proposition 1. *Let v_i be a node of the HEX graph $G = (V, E_h, E_e)$. If $E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_i) = \emptyset$, then there exists $y \in S_G$ such that $y_k = 1, \forall v_k \in \bar{\alpha}(v_i)$ and $y_l = 0, \forall v_l \in V \setminus \bar{\alpha}(v_i)$.*

Proof. We need to show that y is legal, i.e. no constraint is violated. For those labels with value 1, i.e. v_i and its ancestors, there is no violation for hierarchy edges because a violation requires a value 0. Since there is no exclusion edges between them per our assumption, there is no violation on exclusion edges either. For those labels with value 0, there is no violation because any violation requires a value 1. The only possibility of violation is between the labels with value 1 and the labels with value 0, i.e. between v_i plus its ancestors and the rest of the labels. It cannot violate any exclusion edge because violation needs two 1s. So it can only violate hierarchy edges. A violation of a hierarchy edge can only happen if a label with value 0 is an ancestor of a label with value 1, i.e. an ancestor of v_i. But all ancestors of v_i has value 1 so a violation is not possible. □

Theorem 1. *A HEX graph $G = (V, E_h, E_e)$ is consistent if and only if for any label $v_i \in V$, $E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_i) = \emptyset$.*

Proof. We first prove one direction by contradiction. Assume to the contrary that there exists a label $v_i \in V$ such that there exists $(v_k, v_l) \in E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_i)$. Because G has no self loop, $v_k \neq v_l$. For any legal assignment $y \in S_G$, if v_i takes value 1, i.e. $y_i = 1$, all labels in $\bar{\alpha}(v_i)$ must take value 1 because otherwise it causes a violation of the hierarchy edges. This means that v_k and v_l must both take value 1. Since $(v_k, v_l) \in E_e$, this makes y illegal, causing a contradiction.

Now we prove the other direction. Suppose for any label $v_i \in V$, $E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_i) = \emptyset$. We want to show that for any label $v_i \in V$, there exists legal assignments $y, y' \in \{0,1\}^n$ such that $y_i = 1$ and $y'_i = 0$. First it is easy to verify that the assignment $y' = (0, \ldots, 0)$ does not violate any edges, because any violation requires a value 1 on at least one label. So we only need to show that there exists a legal assignment y such that $y_i = 1$. We can construct y by setting $\bar{\alpha}(v_i)$, i.e. y_i and $\alpha(y_i)$, to 1 and setting $V \setminus \bar{\alpha}(v_i)$, i.e. all other labels to 0. Then Proposition 1 implies that y is legal. □

1.2 Efficient Inference

Equivalence Two HEX graphs are *equivalent* if they have the same state space:

Definition 4. *HEX graphs G and G' are equivalent if $S_G = S_{G'}$.*

Definition 5. *Given a graph $G = (V, E_h, E_e)$, a directed edge $e \in V \times V$ (not necessarily in E_h) is redundant if $G' = (V, E_h \setminus \{e\}, E_e)$ and $G'' = (V, E_h \cup \{e\}, E_e)$ are both equivalent to G. An undirected edge $e \in V \times V$ (not necessarily in E_e) is redundant if $G' = (V, E_h, E_e \setminus \{e\})$ and $G'' = (V, E_h, E_e \cup \{e\})$ are both equivalent to G.*

Lemma 1. *Let $G = (V, E_h, E_e)$ be a consistent graph. A directed edge $e \in V \times V$ is redundant if and only if in the subgraph $G = (V, E_h)$ there exists a directed path from v_i to v_j and the path doesn't contain e. An undirected edge $e = (v_i, v_j) \in V \times V$ is redundant if and only if there exists an exclusion edge $e' = (v_k, v_l) \in E_e$ such that $v_k \in \bar{\alpha}(v_i)$, $v_l \in \bar{\alpha}(v_j)$ and $e \neq e'$.*

Proof. We first prove the claim about directed edges. We start with showing that if there exists an alternative path, then the edge is redundant. Obviously adding this edge to a graph without this edge does not enlarge the state space because there are more constraints in the new graph. Also adding the edge does not shrink the state space because every legal assignment in the original state space does not violate this new edge—otherwise it would violate the constraints on the alternative path in the original graph. Now, if a graph has this edge, removing it cannot change the state space because it would contradict with what we just proved—adding the edge back does not change the state space. We now prove the other direction: if the edge (v_i, v_j) is redundant, then there must exists an alternative path from v_i to v_j. Let's denote G as the graph that contains the edge and G' with the edge removed. Assume to the contrary that there is no such a path. Thus v_i is not v_j's ancestor, i.e. $v_i \notin \alpha(v_j)$. Let $y \in \{0,1\}^n$ be an assignment where $(y_i, y_j) = (0,1)$, $y_l = 1, \forall v_l \in \alpha(v_j)$, and $y_k = 0, \forall v_k \in V \setminus (\{v_i, v_j\} \cup \alpha(v_j))$, i.e. we set v_j and its ancestors to 1 and all other nodes to 0. Now assignment y must be a legal assignment of G' according to Proposition 1 because G' is consistent. But y is illegal for G due to the edge (v_i, v_j), which contradicts the assumption that the edge is redundant and G and G' are equivalent.

Now we prove the claim about undirected edges. We start with showing that if there exists an alternative exclusion edge among the node and its ancestors, then the edge is redundant. Obviously adding this edge to a graph without this edge does not enlarge the state space because there are more constraints in the new graph. Also adding this edge does not shrink the state space because every legal assignment in the original state space does not violate this new edge—otherwise both nodes of this new edge as well as the ancestors of both nodes must be 1, violating the alternative exclusion edge. Now, if a graph has this edge, removing it cannot change the state space because it would contradict with what we just proved—adding the edge back does not change the state space. We now prove the other direction: if the edge (v_i, v_j) is redundant, then there exists an alternative exclusion edge between $\bar{\alpha}(v_i)$ and $\bar{\alpha}(v_j)$. Let's denote G as the graph that contains the edge and G' with the edge removed. Assume to the contrary that there is no such an alternative exclusion edge. We now construct an assignment $y \in \{0,1\}^n$ as follows: set $\bar{\alpha}(v_i) \cup \bar{\alpha}(v_j)$ to 1 and set the rest of the nodes to 0. Then y is legal for G' because (1) there are no exclusion edges among $\bar{\alpha}(v_i)$ or among $\bar{\alpha}(v_j)$ as G' is consistent, (2) there are no exclusion edges between $\bar{\alpha}(v_i)$ and $\bar{\alpha}(v_j)$ per our assumption, (3) there are no other violations by the same argument in Proposition 1. But y is illegal for G due to the exclusion edge (v_i, v_j). This contradicts the assumption that G and G' are equivalent. □

Definition 6. *A graph G is* minimally sparse *if it has no redundant edges. A graph G is* maximally dense *if every redundant edge is in G.*

Proposition 2. *For a consistent graph $G = (V, E_h, E_e)$ and any two nodes v_i, v_j, there exists a directed path from v_i to v_j in the subgraph (V, E_h) if and only if the assignment $y \in \{0,1\}^n$, where $y_k = 1, \forall v_k \in \bar{\alpha}(v_j) \setminus \{v_i\}$ and $y_l = 0, \forall y_l \notin \bar{\alpha}(v_j) \setminus \{v_i\}$, is illegal.*

Proof. If there exists a path from v_i to v_j, then it is trivial that the assignment is illegal. We thus prove the other direction. Suppose that there doesn't exists a path from v_i to v_j. Then $\bar{\alpha}(v_j) \setminus \{v_i\} = \bar{\alpha}(v_j)$ and the assignment must be legal because there cannot be any violation between the nodes with value 1 as the graph is consistent. There cannot be any violation between the nodes with value 1 and those with value 0 because it would imply that one ancestor of v_j is assigned value 0. There cannot be a violation between nodes of value 0 because any violation requires at least one value 1. □

Proposition 3. *For a consistent graph $G = (V, E_h, E_e)$ and any two nodes v_i, v_j, there exists an edge $e \in E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_j)$ if and only if the assignment $y \in \{0,1\}^n$, where $y_k = 1, \forall v_k \in \bar{\alpha}(v_i) \cup \bar{\alpha}(v_j)$ and $y_l = 0, \forall y_l \notin \bar{\alpha}(v_i) \cup \bar{\alpha}(v_j)$, is illegal.*

Proof. If there exists an edge $e \in E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_j)$, then it is trivial that the assignment is illegal because the exclusion edge is violated. We thus prove the other direction. Suppose that there doesn't exists an edge $e \in E_e \cap \bar{\alpha}(v_i) \times \bar{\alpha}(v_j)$. Thus the only possible exclusion edges between labels with value 1 are within $\bar{\alpha}(v_i)$ or within $\bar{\alpha}(v_j)$. But this is impossible because the graph is consistent. So there is no violation between labels with value 1. Also there cannot be any violation between the nodes with value 1 and those with value 0 because it would imply that one ancestor of v_j or v_i is assigned value 0. There cannot be a violation between nodes of value 0 because any violation requires at least one value 1. Thus y is legal. □

Proposition 4. *For two consistent and equivalent graphs $G = (V, E_h, E_e)$ and $G' = (V, E'_h, E'_e)$, if there exists a directed path from v_i to v_j in (V, E_h), then must also be a path (not necessarily identical) from v_i to v_j in (V, E'_h).*

Proof. This is a direct consequence of Proposition 2 because otherwise would be an assignment that is legal in G' but illegal in G, contradicting the assumption that G and G' are equivalent. □

Proposition 5. *For two consistent and equivalent graphs $G = (V, E_h, E_e)$ and $G' = (V, E'_h, E'_e)$, if there exists an edge $e \in E_e \cap \bar{\alpha}_G(v_i) \times \bar{\alpha}_G(v_j)$, then there also exists an edge $e' \in E'_e \cap \bar{\alpha}_{G'}(v_i) \times \bar{\alpha}_{G'}(v_j)$.*

Proof. This is a direct consequence of Proposition 3 because otherwise would be an assignment that is legal in G' but illegal in G, contradicting the assumption that G and G' are equivalent. □

Theorem 2. *For any consistent graphs G and G' that are both minimally sparse (or maximally dense), if $S_G = S_{G'}$, then $G = G'$.*

We first prove uniqueness of minimally sparse graphs. Assume to the contrary that $G \neq G'$. Since their state spaces are the same so their nodes must be the same and they can only differ in the edges. We thus show that any difference between edges of the two graphs would lead to a contradiction.

We first consider hierarchy edges. Let $G = (V, E_h, E_e)$ and $G' = (V, E'_h, E'_e)$. Let $e_h = (v_i, v_j) \in E_h$ but $e_h \notin E'_h$, i.e. a hierarchy edge that appears in G but not G'. According to Proposition 4, there must be a path from v_i to v_j in G'. Since the edge (v_i, v_j) is not in G', there must be a path via a third node v_k. Again applying Proposition 4, there thus must be a path from v_i to v_k and a path from v_k to v_j in G as well. The path from v_i to v_k in G cannot contain (v_i, v_j) because otherwise it would mean that there is path from v_j to v_k, forming a loop between v_j and v_k in G. Also, the path from v_k to v_j cannot contain (v_i, v_j) because otherwise it would mean that there is a path from v_k to v_i, forming a loop between v_i and v_k in G. Thus the path from v_i to v_j in G cannot contain the edge $e_h = (v_i, v_j)$, making e_h redundant in G and contradicting the assumption that G is minimally sparse.

We next consider exclusion edges. Let $G = (V, E_h, E_e)$ and $G' = (V, E'_h, E'_e)$. Let $e_e = (v_i, v_j) \in E_e$ but $e_e \notin E'_e$, i.e. an exclusion edge that appears in G but not G'. Per Proposition 5, there must be an exclusion edge $(v_k, v_l) \in E'_e \cap \bar{\alpha}_{G'}(v_i) \times \bar{\alpha}_{G'}(v_j)$ in G'. Per Proposition 4, $v_k \in \bar{\alpha}_G(v_i)$ and $v_l \in \bar{\alpha}_G(v_j)$. Since $e_e \notin E'_e$, thus $(v_k, v_l) \neq (v_i, v_j)$, which implies that (v_i, v_j) is redundant and present in G, which contradicts that G is minimally sparse.

We now prove uniqueness of maximally dense graphs. Let G and G' be consistent, equivalent, and maximally dense. We first show that any hierarchy edge $e_h = (v_i, v_j)$ in G must also be in G'. This is because there must be a directed path from v_i to v_j in G' per Proposition 4. Then $e_h = (v_i, v_j)$ must also be in G' because otherwise e_h is a redundant but absent edge of G', which contradicts the assumption that G' is maximally dense.

We next show that any exclusion edge $e_e = (v_i, v_j)$ in G must also be in G'. First there must be an exclusion edge $e'_e \in E'_e$ between $\bar{\alpha}_{G'}(v_i)$ and $\bar{\alpha}_{G'}(v_j)$ per Proposition 5. If e_e is not in G', then $e_e \neq e'_e$ and e_e is redundant in G', contradicting the assumption that G' is maximally dense. □

Definition 7. *The maximum overlap of a consistent graph $G = (V, E_h, E_e)$ is $\Omega_G = \max_{v \in V} |o_{\bar{G}}(v)|$, where $o_{\bar{G}}(v) = \{u \in V : (u, v) \notin \bar{E}_h \wedge (v, u) \notin \bar{E}_h \wedge (u, v) \notin \bar{E}_e\}$ and $\bar{G} = (V, \bar{E}_h, \bar{E}_e)$ is the maximally dense equivalent of G.*

Proposition 6. *An induced subgraph G' of a consistent graph G is consistent.*

Proof. Assume to the contrary that G' is not consistent. Per Theorem 1, there must exist a node v_i with an exclusion edge among v_i and its ancestors in G'. The same graph structure must also be present in G because G' is an induced subgraph. Thus G is not consistent, which is a contradiction. □

Proposition 7. *An induced subgraph G' of a consistent and maximally dense graph G is also consistent and maximally dense.*

Proof. Per Proposition 6 G' must be consistent. Assume to the contrary that G' is not maximally dense. Then there must be a redundant edge of G' that is absent in G'. This edge must also be redundant and absent in G because the graph structure (Lemma 1) that makes it redundant in G' remains in G. But this is a contradiction. □

Proposition 8. *If G' is an induced subgraph of a consistent, maximally dense graph G, then $\Omega_{G'} \leq \Omega_G$.*

Proof. Per Proposition 6 and 7, G' is also consistent and maximally dense, so we just need to compare $\max_{v \in V'} |o_{G'}(v)|$ and $\max_{v \in V} |o_G(v)|$. Because G' is an induced subgraph of G, every edge not in G' is also not in G. Thus $\forall v \in V', |o_{G'}(v)| \leq |o_G(v)|$. It follows that $\max_{v \in V'} |o_{G'}(v)| \leq \max_{v \in V} |o_G(v)|$ since $V' \subseteq V$. □

Proposition 9. *Let $v_i \in V$ be an arbitrary node of a consistent and maximally dense graph $G = (V, E_h, E_e)$, then V is a disjoint union of $\{v_i\}, \alpha(v_i)$, $\sigma(v_i)$, $\epsilon(v_i)$, and $o(v_i)$.*

Proof. First it is trivial to verify that any node $v \in V$ must belong to at least one of the sets $\{v_i\}, \alpha(v_i), \sigma(v_i), \epsilon(v_i)$, and $o(v_i)$. We only need to show that these sets do not intersect. Obviously $\{v_i\}$ doesn't intersect any of the rest by definition. The ancestors $\alpha(v_i)$ and descendants $\sigma(v_i)$ cannot intersect because they would lead a directed loop. The ancestors cannot intersect with the exclusive nodes $\epsilon(v_i)$ because otherwise it would make G inconsistent per Theorem 1. The ancestors cannot intersect with the overlapping nodes $o(v_i)$ because otherwise there would be a redundant but absent edge per Lemma 1 and G is not maximally dense. The descendants $\sigma(v_i)$ cannot intersect with the exclusive nodes $\epsilon(v_i)$ because otherwise it makes G inconsistent. The descendants $\sigma(v_i)$ cannot intersect with the overlapping nodes $o(v_i)$ because otherwise there would be a redundant but absent edge per Lemma 1 and G is not maximally dense. Finally, the exclusive nodes $\epsilon(v_i)$ cannot intersect with the overlapping nodes $o(v_i)$ by definition. □

Proposition 10. *Let $v_i \in V$ be an arbitrary node of $G = (V, E_h, E_e)$, $V^0 = \alpha(v_i) \cup \epsilon(v_i) \cup o(v_i)$ be the set of ancestors, exclusive nodes and overlapping nodes, and $V^1 = \sigma(v_i) \cup o(v_i)$ be the set of descendants and overlapping nodes.*

Let $G^0 = G[V^0]$ be the subgraph of G induced by V^0. Let $S_G^0 = \{y \in \{0,1\}^n : y_i = 0 \bigwedge y_{\sigma(v_i)} = 0 \bigwedge y_{V^0} \in S_{G^0}\}$, i.e. assignments with node v_i and its descendants set to zero and the rest of values taken from the legal assignments of the subgraph G^0.

Let $G^1 = G[V^1]$ be the subgraph of G induced by V^1. Let $S_G^1 = \{y \in \{0,1\}^n : y_i = 1 \bigwedge y_{\alpha(v_i)} = 1 \bigwedge y_{\epsilon(v_i)} = 0 \bigwedge y_{V^1} \in S_{G^1}\}$, i.e. assignments with node v_i set to 1, its ancestors set to 1, its exclusive nodes set to 0, and the rest of values taken from the legal assignments of the subgraph G^1.

If G is consistent and maximally dense, then $S_G = S_G^0 \cup S_G^1$ and $S_G^0 \cap S_G^1 = \emptyset$.

Proof. It is trivial to show that $S_G^0 \cap S_G^1 = \emptyset$ because the node v_i takes different values. So we only need to show $S_G = S_G^0 \cup S_G^1$.

We first show $S_G \subseteq S_G^0 \cup S_G^1$. Let $y \in S_G$ be a legal assignment. We first consider the case of $y_i = 0$. In this case, all descendants of v_i must take value 0. Per Proposition 9, the rest of the nodes are V^0. Since y is legal for G, thus $y_{V^0} \in S_{G^0}$, i.e. y_{V^0} must be legal for $G[V^0]$ because y_{V_0} cannot violate the constraints in $G[V^0]$, which are a subset of constraints of G. Thus $y \in S_G^0$ if

$y_i = 0$. We now consider the case of $y_i = 1$. In this case, all ancestors must take value 1 and all exclusive nodes must take value 0. Per Proposition 9, the rest of the nodes are V^1. Since y is legal for G, thus $y_{V^1} \in S_{G^1}$, i.e. y_{V^1} must be legal for $G[V^1]$ because y_{V_1} cannot violate the constraints in $G[V^1]$, which are a subset of constraints of G. Thus $y \in S^1_G$ if $y_i = 1$. It then follows that $S_G \subseteq S^0_G \cup S^1_G$.

We now show $S_G \supseteq S^0_G \cup S^1_G$. To do this, we first show $S_G \supseteq S^0_G$ and then show $S_G \supseteq S^1_G$.

Let $y \in S^0_G$. We just need to verify that $y \in S_G$. Since y_{V^0} is legal for $G[V^0]$, y cannot violate any constraints among the nodes of V^0 in G as $G[V^0]$ is an induced subgraph of G. Per Proposition 9, the rest of the nodes $V \setminus V^0$ consists of v_i and its descendants $\sigma(v_i)$. There are no violations of constraints between $V \setminus V^0$ because they are all zero. There are no violations of constraints between $V \setminus V^0$ and V^0 because a violation would require a hierarchy edge from a node in $V \setminus V^0$ to a node in V^0, which is impossible because all descendants of v_i are in $V \setminus V^0$. Thus $y \in S_G$ and $S_G \supseteq S^0_G$.

We now show $S_G \supseteq S^1_G$. Let $y \in S^1_G$. We just need to verify that $y \in S_G$. Since y_{V^1} is legal for $G[V^1]$, y cannot violate any constraints among the nodes of V^1 in G as $G[V^1]$ is an induced subgraph of G. Per Proposition 9, the rest of the nodes $V \setminus V^1$ consists of v_i and its ancestors $\alpha(v_i)$ and exclusive nodes $\epsilon(v_i)$. There are no violations of constraints among $V \setminus V^1$ because all ancestors are 1, all exclusive nodes are 0, and the G is consistent. There are no violations of constraints between $V \setminus V^1$ and V^1 because a violation would require either (1) an exclusion edge (v_i, v_k) for some $v_k \in V^1$, or (2) a hierarchy edge from $\epsilon(v_i)$ to a node $v_k \in V^1$, or (3) an exclusion edge between $\alpha(v_i)$ and a node v_k in V^1. It is easy to verify that case (1) is impossible by definition and case (2) and (3) are also impossible because if such a node $v_k \in V^1$ exists, then (v_i, v_k) is a redundant but absent exclusion edge of G, contradicting that G is maximally dense. Thus $y \in S_G$ and $S_G \supseteq S^1_G$. □

Theorem 3. *For a consistent graph* $G = (V, E_h, E_e)$, $|S_G| \leq (|V| - \Omega_G + 1)2^{\Omega_G}$.

Proof. Let $\bar{G}$ be the maximally dense equivalent of G. Then by definition $\Omega_G = \Omega_{\bar{G}}$ and $S_G = S_{\bar{G}}$. We thus only need to prove this theorem for a consistent and maximally dense G and it then applies for any consistent graph G. So without any loss of generality, we now assume that G is also maximally dense.

We prove by induction. First the claim trivially holds when $G = \emptyset$.

Suppose the claim holds for any $G' = (V', E'_h, E'_e)$ such that $|V'| < |V|$. Let $v_i \in V$ be the node that achieves the maximum overlap in G, i.e.

$$\Omega_G = |o_G(v_i)|. \tag{1}$$

Given v_i, let $V^0, V^1, G^0, G^1, S^0_G, S^1_G$ be the same as defined in Proposition 10. Since $|V^0| < |V|$ and $|V^1| < |V|$ as v_i is excluded from both, the induction hypothesis implies that

$$\begin{aligned}
|S_{G^0}| &\leq (|V^0| - \Omega_{G^0} + 1)2^{\Omega G^0} \\
&= (|\alpha_G(v_i)| + |\epsilon_G(v_i)| + |o_G(v_i)| - \Omega_{G^0} + 1)2^{\Omega G^0} \text{ (Proposition 9)} \\
&\leq (|\alpha_G(v_i)| + |\epsilon_G(v_i)| + 1)2^{\Omega_G} \text{ (Proposition 8)},
\end{aligned}$$

Algorithm 1 Listing state space

function ListStateSpace(graph G)
  if $G = \emptyset$ **then return** $\emptyset$
  end if
  Let $G = (V, E_h, E_e)$ and $n = |V|$.
  Pick an arbitrary $v_i \in V$.
  $V^0 \leftarrow \alpha(v_i) \cup \epsilon(v_i) \cup o(v_i)$.
  $G^0 \leftarrow G[V^0]$
  $S_{G^0} \leftarrow$ ListStateSpace(G^0)
  $S_G^0 \leftarrow \{y \in \{0,1\}^n : y_i = 0 \bigwedge y_{\sigma(v_i)} = 0 \bigwedge y_{V^0} \in S_{G^0}\}$.
  $V^1 \leftarrow \sigma(v_i) \cup o(v_i)$.
  $G^1 \leftarrow G[V^1]$
  $S_{G^1} \leftarrow$ ListStateSpace(G^1)
  $S_G^1 = \{y \in \{0,1\}^n : y_i = 1 \bigwedge y_{\alpha(v_i)} = 1 \bigwedge \epsilon(v_i) = 0 \bigwedge y_{V^1} \in S_{G^1}\}$
  return $S_G^0 \cup S_G^1$.
end function

and

$$\begin{aligned}|S_{G^1}| &\leq (|V^1| - \Omega_{G^1} + 1)2^{\Omega G^1} \\ &= (|\sigma_G(v_i)| + |o_G(v_i)| - \Omega_{G^1} + 1)2^{\Omega G^1} \text{ (Proposition 9)} \\ &\leq (|\sigma_G(v_i)| + 1)2^{\Omega_G} \text{ (Proposition 8).}\end{aligned}$$

Per Proposition 10,

$$\begin{aligned}|S_G| &= |S_G^0| + |S_G^1| \\ &= |S_{G^0}| + |S_{G^1}| \\ &\leq (|\alpha_G(v_i)| + |\epsilon_G(v_i)| + 1)2^{\Omega_G} + (|\sigma_G(v_i)| + 1)2^{\Omega_G} \\ &= (|\alpha_G(v_i)| + |\epsilon_G(v_i)| + |\sigma_G(v_i)| + 2)2^{\Omega_G} \\ &= (|\alpha_G(v_i)| + |\epsilon_G(v_i)| + |\sigma_G(v_i)| + |o_G(v_i)| + 1 - |o_G(v_i)| + 1)2^{\Omega_G} \\ &= (|V| - \Omega_G + 1)2^{\Omega_G} \text{ (Proposition 9 and Eqn. 1).}\end{aligned}$$

□

Lemma 2. *If graph $G = (V, E_h, E_e)$ is consistent and maximally dense, then Algorithm 1 runs in $O((|V| + |E_h| + |E_e|)|S_G|)$. time and returns the state space S_G.*

Proof. We first prove by induction the correctness of Algorithm 1, i.e. it returns S_G. If the graph is empty, then the correctness trivially holds. Suppose the claim holds for any graph $G' = (V', E_h', E_e')$ where $|V'| < |V|$. Since $|V^0| < |V|$ and $V^1| < |V|$ as v_i is excluded, S_{G^0} and S_{G^1} in Algorithm 1 are the state spaces for G^0 and G^1 per the induction hypothesis. Then Proposition 10 implies that the algorithm returns S_G.

We now prove that the running time $T(G)$ is $O(|S_G|)$. First it is easy to verify that there exists a constant $c > 0$ such that for any graph G Algorithm 1 can be implemented such that

$$T(\emptyset) = c,$$

and

$$T(G) \leq c(|V| + |E_h| + |E_e|) \text{ (Line 2-7, Line 10,11)} \\ +T(G^0) + T(G^1) \text{ (Line 8,12).}$$

Note that Line 9, 13, 14 can be implemented with a cost $O(|V|)$ (i.e. independent of $|S_G^0|$ and $|S_G^1|$) because we can return the state space as a binary tree instead of a list. In this case, Line 2 returns one empty node. Line 9 (Line 13) fills its padding values into the root node of the binary tree returned at Line 8 (Line 12). Line 14 creates a new node and links the binary tree returned at Line 8 (Line 12) as the left (right) child. To generate a list of the state space of G, we simply traverse the binary tree and the cost is $O(|V||S_G|)$, which is the lower bound because the lengh of each assignment is $|V|$.

We prove by induction the following claim: for any graph $G \neq \emptyset$,

$$T(G) \leq c(|V| + |E_h| + |E_e|)(2|S_G| - 1). \tag{2}$$

If $|V| = 1$, the claim is trivially true as G^0 and G^1 are empty. Suppose the claim holds for any graph $G' = (V', E'_h, E'_e)$ where $1 \leq |V'| < |V|$. Since $|V^0| < |V|$ and $V^1| < |V|$ as v_i is excluded,

$$\begin{aligned} T(G) &\leq c(|V| + |E_h| + |E_e|) + T(G^0) + T(G^1) \\ &\leq c(|V| + |E_h| + |E_e|) \\ &\quad +c(|V^0| + |E_h^0| + |E_e^0|)(2|S_{G^0}| - 1) \\ &\quad +c(|V^1| + |E_h^1| + |E_e^1|)(2|S_{G^1}| - 1) \\ &\leq c(|V| + |E_h| + |E_e|)(2|S_{G^0}| + 2|S_{G^1}| - 1) \\ &= c(|V| + |E_h| + |E_e|)(2|S_G| - 1) \text{ (Proposition 10).} \end{aligned}$$

Thus $T(G) = O\left((|V| + |E_h| + |E_e|)|S_G|\right)$. □

Lemma 3. *If G' is an induced subgraph of a consistent, maximally dense graph G, then $|S_G| \geq |S_{G'}|$.*

Remark 1. This lemma states that any subgraph induced on a consistent and maximally dense graph has an equal or smaller state space. This fact may look trivial but is not true for arbitrary graphs as a subgraph may have fewer constraints than a full graph. See Sec. 2.3 in this supplemental material for counter-examples.

Proof. Let $G = (V, E_h, E_e)$ and $G' = (V', E'_h, E'_e)$. Let y' be an arbitrary legal assignment $y' \in S_{G'}$. Let $V^1 = \{v_i \in V' : y'_i = 1\}$ and $V^0 = \{v_i \in V' : y'_i = 0\}$. Let $A = \cup_i \bar{\alpha}_G(v_i)$. We construct a new assignment y such that $y_A = 1$ and $y_{V \setminus A} = 0$, i.e. setting all nodes in A to 1 and the rest to 0.

We first show that $y_{V'} = y'$ because (1) $y_{V^1} = 1$ since $V^1 \subseteq A$ by definition and (2) $y_{V^0} = 0$ since otherwise there exists a node $v_k \in V^0$ (i.e. $y'_k = 0$) such

Algorithm 2 Exact Inference

Input: Graph $G = (V, E_h, E_e)$.
Input: Scores $f \in \mathcal{R}^{|V|}$.
Output: Marginals, e.g. $\Pr(v_i = 1)$.
1: $G^* \leftarrow$ SPARSIFY(G)
2: $\bar{G} \leftarrow$ DENSIFY(G)
3: $T \leftarrow$BUILDJUNCTIONTREE(G^*).
4: For each clique $c \in T$, $S_c \leftarrow$LISTSTATESPACE($\bar{G}[c]$).
5: Perform (two passes) message passing on T using only states S_c for each clique c.

that $y_k = 1$, i.e. $v_k \in \bar{\alpha}_G(v_l)$ for some $v_l \in V^1$ (i.e. $y'_l = 1$), meaning that (v_k, v_l) is redundant in G per Lemma 1 and must be present in G per the fact that G is maximally dense, further implying that $(v_k, v_l) \in E'_h$ with $(y'_k, y'_l) = (0, 1)$, which contradicts the assumption that $y' \in S_{G'}$. Thus a constructed assignment based on y' will be different if y' is different. Thus to prove the claim, we just need to show that every constructed assignment is legal for G.

Next we show that there cannot be any constraint violations for G between nodes in A. Assume to the contrary that an exclusion edge (v_i, v_j) is violated. Now consider $\bar{\sigma}_G(v_i) \cap V^1$ and $\bar{\sigma}_G(v_j) \cap V^1$, both of which cannot be empty per the definition of A. If $\bar{\sigma}_G(v_i) \cap V^1 = \bar{\sigma}_G(v_j) \cap V^1$ it causes a contraction with the assumption that G is consistent. If $\bar{\sigma}_G(v_i) \cap V^1 \neq \bar{\sigma}_G(v_j) \cap V^1$, then there exist $v_k \in \sigma_G(v_i) \cap V^1$ and $v_l \in \sigma_G(v_j) \cap V^1$ such that $v_k \neq v_l$, which means that the undirected edge (v_k, v_l) is redundant and must be present in G since G is maximally dense. Thus the exclusion edge (v_k, v_l) must also be present in G'. This contracts with the assumption that $y' \in S_{G'}$.

We also show that there cannot be any constraint violations for G between A and $V \setminus A$, because otherwise there exists a hierarchy edge $(v_i, v_j) \in E_h$ where $v_i \in V \setminus A$ and $v_j \in A$, implying that $v_i \in A$, which contradicts that $v_i \in V \setminus A$. Finally there cannot be any violation among $V \setminus A$ because all values are 0. □

Theorem 4. *The complexity of the exact inference (Line 5 in Algorithm 2) for graph $G = (V, E_h, E_e)$ is $O\left(\min\{|V|2^w, |V|^2 2^{\Omega_G}\}\right)$, where w is the width of the junction tree T (Line 3).*

Proof. The computation at a clique c is $O(|V|2^{\Omega_G})$ because $|S_c| \leq |S_G| \leq (|V| + 1)\Omega_G$ per Lemma 3 and Theorem 3. It is a standard result that a junction without redundancy has at most $|V|$ cliques (redundant cliques can simply be removed). Thus the total computation is $O(|V|^2 2^{\Omega_G})$. At the same time, standard results on junction trees show that the total computation is $O(|V|2^w)$. □

Remark 2. The bound in Theorem 4 is a worst case bound as it does not make assumptions on how the junction tree is built. If in choosing a junction tree we always include as a candidate the junction tree with only one clique, then the bound can be improved to $O\left(\min\{|V|2^w, |V|2^{\Omega_G}\}\right)$ per Theorem 3.

2 Discussions

In this section we discuss certain technical issues in more detail.

2.1 The importance of consistency

Consistency is important because inconsistent graphs will break some key results that our algorithms rely on.

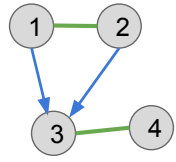

Fig. 1. An inconsistent HEX graph where Lemma 1 does not hold.

For example, consider Fig. 1. Here v_3 is a "dead" node, thus making the edge (v_3, v_4) redundant because v_3 is always zero and cannot conflict with v_4. But the graph structure stated in Lemma 1 cannot detect this redundant edge.

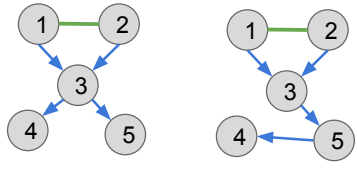

Fig. 2. Equivalent, minimally sparse but inconsistent HEX graphs where Theorem 2 does not hold.

Another example is in Fig. 2. Here v_3, v_4, v_5 are always 0 due to the inconsistency. It is easy to verify that the two graphs are equivalent. Also both graphs are minimally sparse because no edge can be removed without changing the state space. However, the two graphs are not the same and thus Theorem 2 does not apply.

2.2 Softmax as a special case

If we use a HEX graph with pairwise exclusion edges and no hierarchy edges, i.e. all nodes are mutually exclusive, it is easy to verify that our classification

model is equivalent to the popular softmax (or multinomial regression) up to an additional constant 1 in the denominator.

$$\Pr(y_i = 1|x) = e^{f_i}/(1 + \sum_j e^{f_j})$$

This additional constant corresponds to the one extra class "none of the above". It makes no practical difference because we can simply add a fixed bias to each score and the constant will effectively go away as the bias approaches infinity. In addition, in a standard multiclass setting we never observe the class "none of the above" during training and testing. This additional constant thus has no effect on the optimal decision boundary because the prediction only depends on the relative strength of the scores. Empirically we have observed no noticeable change of results in our experiments.

2.3 Counter-example of Lemma 3 under different conditions

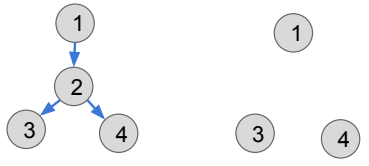

Fig. 3. A consistent but not maximally dense graph where Lemma 3 does not hold: the subgraph induced by v_1, v_3, v_4 have more legal states.

Fig. 3 gives a counter example of Lemma 3 when not all of its conditions are met. Here the graph is consistent but not maximally dense. It is easy to verify that the legal states of the full graph are $\{0000, 1000, 1100, 1101, 1110, 1111\}$, a total of 6 legal states. But the subgraph induced by nodes v_1, v_3, v_4 has no edges and thus has a total of 8 legal states.