Notes on Awasthi Et Al. (2015)

Karl Stratos

1 Setting

1.1 Clustering Error

Fix data $X = \{x_1 \dots x_n\}$. A k-clustering $C = \{c_1 \dots c_k\}$ is a partition of X. An asymmetric distance between clusterings C, C' is

$$\text{dist}\,(C, C') := \sum_{c \in C} \text{dist}\,(c, C')$$

where $\text{dist}\,(c, C')$ counts how many *extraneous* clusters in C' overlap with c,

$$\text{dist}\,(c, C') := |\{c' \in C' : c \cap c' \neq \varnothing\}| - 1$$

Let C^* denote the ground-truth clustering and C our hypothesis. We consider two types of error.

Overclustering error. $\delta_o := \text{dist}\,(C, C^*)$

- $\delta_o = 0$ iff every hypothesis cluster is contained in a single gold cluster.
- $\delta_o > 0$ iff some hypothesis cluster $c \in C$ overlaps with multiple gold clusters.
  - c is "overclustering" and needs to be split.

Underclustering error. $\delta_u := \text{dist}\,(C^*, C)$

- $\delta_u = 0$ iff every gold cluster is contained in a single hypothesis cluster.
- $\delta_u > 0$ iff multiple hypothesis clusters $c, c' \in C$ overlap with a gold cluster.
  - c, c' are "underclustering" and needs to be merged.

Note that $C = C^*$ iff $\delta_o = \delta_u = 0$.

1.2 Consistent Global Tree

We use a tree T over X to efficiently compute local edits on C that are consistent with C^*. The condition on the tree is the following.

Consistency condition. Pick any node N in T. If a gold cluster $c^* \in C^*$ overlaps with N, either $c^* \subseteq N$ or $c^* \supseteq N$. (Thus C^* is a k-pruning of T.)

The paper derives this condition by assuming "stability" condition on C^*: if $A \subset c^*$, then for any $c^+ \neq c^*$

$$S(A, c^* \backslash A) > S(A, c^+)$$

where S is the expected similarity between subsets of X. If we use an average-linkage tree, then it must be consistent with C^*. At any point in constructing the tree, we seek a pair of partial clusters $A, B \subset X$ with maximum $S(A, B)$, and by the stability condition they must belong to the same ground-truth cluster.

Remark. This requirement reduces the problem to **finding a correct pruning** of T, a much weaker result than what the paper makes it sound like. Can we split/merge clusters *without* T, just by halving and combining? In fact, these tree-less edits might even be better when T is misleading!

2 Algorithm

2.1 The η-Merge Model

An "oracle" issues these requests on C until it's equal to $C^* = \{c_1^* \dots c_k^*\}$.

- `split`(c) is issued when there are $x, x' \in c$ such that $x \in c_i^*$ and $x' \in c_j^*$ for $i \neq j$.
- `merge`(c, c') is issued when there is some $i \in [k]$ such that $|c \cap c_i^*| \geq \eta |c|$ and $|c' \cap c_i^*| \geq \eta |c'|$.

The merge operation is a generalization of Balcan and Blum (2008): setting $\eta = 1$ ensures that clusters must be pure before a merge.

2.2 Split

Upon `split`(c), we look up the global tree T and find a node that divides the points in c. If we choose the *top-most* such node, it covers all points in c, so we can split it into c_1 and c_2 accordingly. This always results in a "clean split": if for some $i \in [k]$ we have $J := c_i^* \cap c \neq \varnothing$, then either $J \subseteq c_1$ or $J \subseteq c_2$ (otherwise it violates the consistency condition on T).

Lemma 2.1. *A clean split reduces the overclustering error δ_o by 1.*

Proof. Let k and k_i denote the number of gold clusters overlapping with c and c_i. In a clean split, $k = k_1 + k_2$. Therefore,

$$\delta_o' = \delta_o - (k - 1) + (k_1 - 1) + (k_2 - 1) = \delta_o - 1$$

where δ_o and δ_o' denote the overclustering error before and after the split operation. □

2.3 Merge

$\texttt{merge}(c, c')$ means some $c^* \in C^*$ overlaps η-significantly with (c, c'). Let N^* denote the node in T that corresponds to c^*; it is one of the nodes in T that overlap η-significantly with (c, c'),

$$S_{c,c'} := \{N \in T : \ |N \cap c| \geq \eta\,|c|\,, \ \ |N \cap c'| \geq \eta\,|c'|\}$$

How can we ensure that a merge results in a cluster "pure" in c^*? This is achieved by first noting that if $\eta > 1/2$, then any $N, N' \in Y$ have an ascendant-descendant relationship. Otherwise, we have $N \cap N' = \varnothing$ but each contains more than a half of the points in (c, c'), a contradiction. Thus, if we pick the *deepest* node $N_{deepest} \in Y$, then we ensure that $N_{deepest} \subseteq N^*$. So we can

- Introduce a new cluster $c'' = N_{deepest} \cap (c \cup c')$ that's purely in c^*, and
- Deplete $c \leftarrow c \backslash N_{deepest}$ and $c' \leftarrow c' \backslash N_{deepest}$.

Additionally, we can mark c'' as "pure" for later. It will never be split, and the next time it's used in a merge it will be completely depleted by setting the corresponding η to one. For example, upon $\texttt{merge}(c, c')$, if c is impure and c' is pure, then we find the deepest node $N_{deepest}$ in

$$S_{c,c'} := \{N \in T : \ |N \cap c| \geq \eta\,|c|\,, \ \ |N \cap c'| \geq |c'|\}$$

and $c' \leftarrow c' \backslash N_{deepest} = \varnothing$. In the beginning of the algorithm, every $c \in C$ is labeled impure.

Lemma 2.2. *There can be at most $2(\delta_u + k) \log_{1/(1-\eta)} n$ merges, where δ_u is the initial underclustering error and n is the number of data points.*

Proof. The number of $\texttt{merge}(c, c')$ when (c, c') are both pure is bounded by the number of merges involving an impure cluster because one such merge creates one pure cluster. We show the latter number is bounded by $(\delta_u + k) \log_{1/(1-\eta)} n$ to prove the lemma.

We examine overlaps between impure clusters and gold clusters:

$$P = \{c \cap c^* : \ c \in C \text{ is impure, } \ c^* \in C^* \text{ overlaps with } c \ \}$$

Initially every $c \in C$ is impure, so $|P| = \sum_{c^*} |\{c : \ c \cap c^* \neq \varnothing\}| = \delta_u + k$. Merge cannot increase $|P|$ since a new cluster is always pure. Split does not change $|P|$ (otherwise it's not a clean split). Now, each merge depletes some $p \in P$ by η portion, and it is sufficient to take $t = \log_{1/(1-\eta)} |p| \leq \log_{1/(1-\eta)} n$ merges to achieve $(1 - \eta)^t\,|p| = 1$ (at which point p will be pure). Thus the total number of impure merges is bounded by $(\delta_u + k) \log_{1/(1-\eta)} n$. □