Article

https://doi.org/10.1038/s41588-024-01993-3

Consensus prediction of cell type labels in single-cell data with popV

Received: 2 September 2023

Accepted: 18 October 2024

Published online: 20 November 2024

Check for updates

Can Ergen[1,2], Galen Xing[1,3,7], Chenling Xu[1], Martin Kim[2], Michael Jayasuriya[2], Erin McGeever[3], Angela Oliveira Pisco[3,8], Aaron Streets[1,3,4] & Nir Yosef[1,2,5,6] ✉

Cell-type classification is a crucial step in single-cell sequencing analysis. Various methods have been proposed for transferring a cell-type label from an annotated reference atlas to unannotated query datasets. Existing methods for transferring cell-type labels lack proper uncertainty estimation for the resulting annotations, limiting interpretability and usefulness. To address this, we propose popular Vote (popV), an ensemble of prediction models with an ontology-based voting scheme. PopV achieves accurate cell-type labeling and provides uncertainty scores. In multiple case studies, popV confidently annotates the majority of cells while highlighting cell populations that are challenging to annotate by label transfer. This additional step helps to reduce the load of manual inspection, which is often a necessary component of the annotation process, and enables one to focus on the most problematic parts of the annotation, streamlining the overall annotation process.

Cell-type annotation is a crucial task in analyzing single-cell RNA sequencing (scRNA-seq) data. The quality of the annotations has a direct impact on downstream analyses such as the comparison of cell type composition as well as the analysis performed on a per-cell-type basis[1]. Manual annotation is highly time-consuming and requires biological context-specific and sequencing technology-specific domain knowledge. Thus, as scRNA-seq becomes an increasingly standard lab technique, there is a growing need to generate automated annotations. We propose here the use of a collection of cell-type prediction models to provide not only automated annotations but also well-calibrated measures of uncertainty. This enables the user to streamline the annotation process.

Automated cell type annotations encounter several challenges[2]. There is no gold standard ground truth for cell type annotation within a specific dataset. Biology is complex, and when cell states vary continuously, delineations between cell types are imprecise, and even human experts may disagree on the exact phenotype of a specific cell. Therefore, it is essential that annotation methods highlight areas of uncertainty that require expert knowledge input. The continuous nature of cell states[3], along with stochasticity in the sequencing process, as well as the domain knowledge of the person manually annotating the dataset, can lead to cells being annotated at varying levels of specificity even within the same dataset. Across multiple datasets, factors, like identification of new cell subtypes or redefinition of marker genes, lead to discrepancies in cell type identification. There are a plethora of automated cell-type annotation methods[4]. However, differences in cell type granularity, experiment-specific nuisance factors and technology-dependent sparsity of gene expression lead to no clear 'best method' for automatic annotation. Based on these factors, we propose that it is crucial for automatic cell-type annotation pipelines[5] to highlight areas of uncertainty that may require manual scrutiny, balance the specificity of predictions with accuracy and be easily accessible and usable.

To address these challenges, we developed popular Vote (popV), a flexible and scalable automated cell-type annotation framework that takes in an unannotated query dataset from a scRNA-seq experiment,

[1]Center for Computational Biology, University of California, Berkeley, Berkeley, CA, USA. [2]Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, Berkeley, CA, USA. [3]Chan Zuckerberg Biohub—San Francisco, San Francisco, CA, USA. [4]Department of Bioengineering, University of California, Berkeley, Berkeley, CA, USA. [5]System Immunology, Weizmann Institute of Science, Rehovot, Israel. [6]Ragon Institute of MGH, MIT, and Harvard, Cambridge, MA, USA. [7]Present address: Gladstone Institute of Genomic Immunology, San Francisco, CA, USA. [8]Present address: Insitro, South San Francisco, CA, USA. ✉e-mail: nir.yosef@weizmann.ac.il

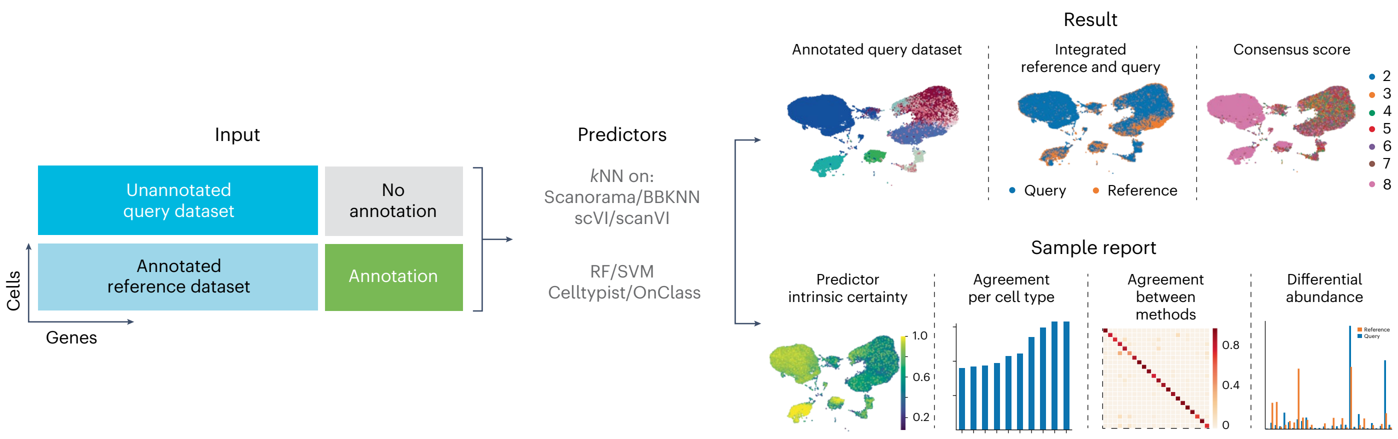

Input
Unannotated query dataset
No annotation
Annotated reference dataset
Annotation
Cells
Genes
Predictors
kNN on: Scanorama/BBKNN scVI/scanVI
RF/SVM Celltypist/OnClass
Result
Annotated query dataset
Integrated reference and query
Consensus score
Query
Reference
Sample report
Predictor intrinsic certainty
Agreement per cell type
Agreement between methods
Differential abundance

Fig. 1 | Framework of popV for automatic cell type annotation. PopV takes an unannotated query dataset and an annotated reference dataset as input. Each expert algorithm predicts the label on the query dataset to yield a cell-type annotation. The certainty of the respective label transfer can be quantified by scoring the agreement of those methods. The workflow yields a sample report to provide the user with insights into the annotated labels.

transfers labels from an annotated reference dataset and generates predictions with a predictability score indicating the confidence of the prediction. We pose here that various prediction methods will disagree in their prediction if an annotation is not accurate, whereas they will tend to agree if the predicted cell type is the correct one. We named our method popV because instead of relying on the predictions of a single classifier, popV takes a consensus approach and incorporates the predictions from eight automated annotation methods. PopV also takes into account annotations at different levels of granularity by aggregating results over the Cell Ontology[6], an expert-curated formalization of cell types in a hierarchical structure with a standardized vocabulary.

PopV is available as an easy-to-install, open-source Python package and is designed to be a flexible framework for incorporating future cell-type classification methods. We provide a notebook that allows the prediction of new datasets and provides pretrained models for 20 different organs based on the Tabula Sapiens dataset[7].

Results

Overview of popV

PopV takes a consensus of experts' approach to the task of automated cell type annotation. The input is an unannotated query dataset together with an annotated reference dataset (Fig. 1a). Both datasets are expected to contain raw count data and demonstrate that popV can be applied to unique molecular identifier as well as non-unique molecular identifier-based technologies. PopV then runs the following eight different annotation methods: random forest (RF)[8], support vector machine (SVM)[8], scANVI[9], OnClass[10], Celltypist[11] and *k*-nearest neighbors (*k*NN) after batch correction with three single-cell harmonization methods—scVI[12], BBKNN[13] and Scanorama[14] (Fig. 1). The eight prediction algorithms were chosen because they were shown to have good prediction accuracy[15] and/or good harmonization performances[16]. These methods encompass supervised methods that are trained only on labeled data (RF, SVM, OnClass, Celltypist and *k*NN) after applying unsupervised harmonization methods that are agnostic to label information during training (BBKNN, Scanorama and scVI) and a semi-supervised method trained with both labeled and unlabeled data (scANVI). However, we emphasize that popV offers an intuitive application interface (API) for the rapid inclusion of additional annotation methods. We demonstrate this capability through a code snippet for adding a new classifier (*k*NN after batch correction with Harmony[17]) in the Methods.

After applying each of these methods separately, popV proceeds to aggregate the resulting predictions for two purposes (Extended Data Fig. 1). The first is to designate a single 'consensus' annotation for every query cell. The second purpose is to quantify our certainty in this prediction. We estimate the consensus annotation using a simple majority vote procedure, counting for each annotation label the number of algorithms that support it. In this procedure, all algorithms get a single 'vote', except for OnClass, which received several votes. The reason for that is that OnClass is the only method in our collection of methods that is capable of predicting cell types that do not exist in the reference dataset. It does so through a two-step process—first selecting an annotation out of the collection of labels in the reference dataset and then propagating it to identify a potentially more refined label in the Cell Ontology (even if this label is absent from the reference). To account for these 'out of sample' cell type annotations, we consider every label that is on the path from the root of the ontology down to the OnClass-predicted label as a predicted label (Extended Data Fig. 1). We then perform majority voting with OnClass having multiple 'votes' at different levels of hierarchy. We have attempted using a simple majority vote with the 'within sample' annotation from the first stage of OnClass and with no propagation along the Cell Ontology. In most of our analyses, we found our first strategy to outperform the simple strategy (Supplementary Fig. 1).

A potentially useful property of many of the algorithms included in popV is an 'algorithm-intrinsic' estimation of prediction certainty. This could, in principle, be leveraged to compute a weighted consensus. However, we found that the certainties are calibrated differently for the different methods, which makes this approach futile as it will weigh more on the predictions of classifiers with higher estimated certainties.

After calculating the consensus score, popV generates a sample report that includes prediction summaries as well as integrated views of the query and reference datasets. For the latter, it displays Uniform Manifold Approximation and Projections (UMAPs) for the joint visualization of the reference and query datasets for the four methods that perform data integration (Fig. 1), as well as a bar plot comparing cell type frequencies in the reference and query dataset to highlight the differential abundance of various cell types. One set of summaries in the report is confusion matrices between the consensus predictions and each individual method to indicate which cell types were confused with another cell type for any particular method. The report also includes a per-cell-type display of the consensus score (that is, the number of agreeing methods—between 1 and 8) to highlight which cell types are overall difficult to predict. Complementing this 'algorithm-extrinsic' estimation of certainty, we also output the intrinsic uncertainty (that is, classifier score) of each of the eight methods (these scores are defined in the Methods). We emphasize that intrinsic and extrinsic uncertainty are two complementary measurements essential to quantifying the performance of a set of cell annotation tools.

To allow for fast annotation of new query datasets, we provide pretrained models for all 20 organs present in Tabula Sapiens[18]. Pretraining is possible for all methods except Scanorama and BBKNN, which compute a joint embedding of reference and query datasets and make pretraining infeasible. For scVI and scANVI, we provide pretrained embeddings of the reference dataset and map the query dataset to this embedding using scArches[19]. PopV has the following three different modes of cell type prediction: in retrain mode, all classifiers are trained from scratch, which requires an hour for 100k cells in a Google Colab session; in inference mode, models previously trained for the reference dataset are used where applicable, which requires 30 min for 100k cells; in fast mode, only pretrained models are used and only cell types in the query dataset are predicted, which requires 5 min for 100k query cells. PopV is available as an open-source Python project and includes an online Google Colab notebook with free computing resources. The codebase enables the addition of new reference datasets (in addition to Tabula Sapiens) through a simple API and can be invoked from the same notebook environment. We recommend that in any newly added reference datasets, the annotations should be consistent with Cell Ontology, either by matching terms in the ontology or by hierarchically assigning new terms to existing terms in the ontology. To this end, we provide scripts to add custom cell type labels to the Cell Ontology before processing by popV (where it is used for running OnClass and calculating our consensus scores).

PopV prediction score discriminates high- and low-quality annotations

We evaluated the performance of cell-type annotation using popV with a Human Lung Cell Atlas as the query dataset[20] and the lung tissue of Tabula Sapiens as a reference dataset. The Lung Cell Atlas is carefully annotated to a high level of granularity. It contains a wide variety of cell types across immune cells, epithelial cells, endothelial cells and stromal cells and is therefore well suited for studying tissues with diverse labels. To make the labels comparable across both datasets, we translated the Lung Cell Atlas labels to the corresponding terms in the Cell Ontology (Supplementary Fig. 2).

PopV achieves high accuracy on the Lung Cell Atlas. We visualize the popV predictions against the manual annotations in the Lung Cell Atlas and see a strong agreement between the prediction and the original annotation, as well as a good integration between the query and the reference cells (Fig. 2a). We decided here to use scANVI integration as it showed the highest performance in scIB metrics, which measure data integration and biological conservation[16] (Extended Data Fig. 2a). To evaluate the quality of our predictions, we compute accuracy terms based on the Cell Ontology tree (Methods). An exact match, as the name implies, means that the predicted cell type is exactly the same as the manual annotation. Furthermore, intuitively, a prediction algorithm that predicts one cell type as another similar cell type performs better than a prediction algorithm that predicts the cell is of an unrelated type. The parent match, child match and sibling match take this into account and measure if the predicted cell type is the parent, child or sibling in the Cell Ontology tree compared to the ground truth annotation. This measure is especially useful if a cell type label exists only in the query and not in the reference dataset. Every prediction that did not match any of these relationships was classified as no match. PopV overall achieves high accuracy for most cell types (Fig. 2b and Extended Data Fig. 2c). Except for scANVI and OnClass, all methods have comparable performance in this dataset. Furthermore, we compared the performance of popV with the label transfer provided in Seurat, which is another popular tool for cell type annotation transfer[21], and found that Seurat performs worse than most methods used in popV. We also included OnClass predictions after step one (OnClass_seen), where OnClass only predicts cell types that were present in the reference dataset, and found this to perform similarly to the good-performing annotation tools, so that the lower performance of OnClass here is solely due to the prediction of unseen cell types. Overall, popV performed best for the number of exact matches and was comparable in the number of cells with no match, highlighting that the popV prediction is more accurate than any of the single methods. For a better insight into the prediction, we display bar plots in the report for popV, highlighting the abundance of cell types in query and reference datasets, as well as prediction accuracy (Extended Data Fig. 2), and display confusion of cell types using alluvial plots (Supplementary Fig. 3).

When checking the popV prediction scores, we found that the accuracy of the prediction is highly correlated with the prediction score (Fig. 2c). For scores of 6 and higher, we found that more than 90% of the annotations were exact matches with the ground truth. For scores of 8, which is a perfect agreement between all methods, 98% of the predictions were exact matches. For scores of 3 and lower, the prediction accuracy was lower than 50%, highlighting that the popV consensus score is a valuable metric to reflect the classification accuracy and points to groups of cells that should be further (and manually) scrutinized.

When considering cells that were assigned with a low consensus score, we found three possible reasons that may explain the disagreement between the different methods (Fig. 2d). The first is that the distinction between certain cell subsets with different labels is unclear. This often arises in cases of a continuum of cell states with no clear decision boundary in transcriptome space. In such cases, the boundaries determined by different algorithms may vary (because they depend on different objectives or techniques), leading to low consistency. It is, however, exactly those cases that merit closer (and often manual) inspection and—if needed—assignment of multiple optional labels. As an example, we found several areas of low consensus score in the various lung endothelial cells (Fig. 2e). Most endothelial cells with a low consensus score arise between capillary endothelial cells and alveolar capillary type 2 endothelial cells. In this region, the various algorithms disagree on the correct boundary, but all algorithms predict those cells with either of those labels. We found that alveolar capillary type 2 endothelial cells express *EDNRB* and *HPGD*, and capillary endothelial cells express *FCN3* and *IL7R*. Cells between both cell types are double positive in both markers, while they do not show any specific marker gene. Therefore, we conclude that neither the term capillary endothelial cell nor the term alveolar capillary type 2 endothelial cell is adequate to describe these cells, but their phenotype is between both cell types. Thus, it is a region that requires manual scrutiny to determine the correct label of those cells. In fact, such scrutiny was applied in the original annotation of the Lung Cell Atlas—annotations not provided to popV—which labeled these cells as capillary intermediates 1 and 2. Therefore, this example demonstrates that a low consensus score can help identify areas that require a refined label, possibly extending the vocabulary available in the reference atlas.

The second reason for a low consensus score in this case study occurs when the query dataset contains subsets of cells that are absent from the reference atlas. As an example, while the Lung Cell Atlas (which we use as the query) includes a subset of endothelial cells that were originally labeled bronchial vessel 2, this subset (and its respective label) seems to be absent from our reference atlas. Indeed, when checking marker genes for these cells, their expression was high in *PLVAP* and low in the vein endothelial marker *ACKR1* (Supplementary Fig. 4), which can be interpreted as an intermediate stage between capillary endothelial cells (negative for both markers) and lung microvascular endothelial cells (positive for both markers). This combination of marker gene expression was not observed in Tabula Sapiens and therefore marks a cell type not present in the reference dataset.

The third possible reason we find for the low consensus is inaccuracy in the reference annotation. As an example, we found a subset of T cells with a low consensus score (Extended Data Fig. 3). All cells of this group in the query dataset were originally labeled (by the authors of the Lung Cell Atlas) as effector CD4$^+$ αβ T cells, while similar cells

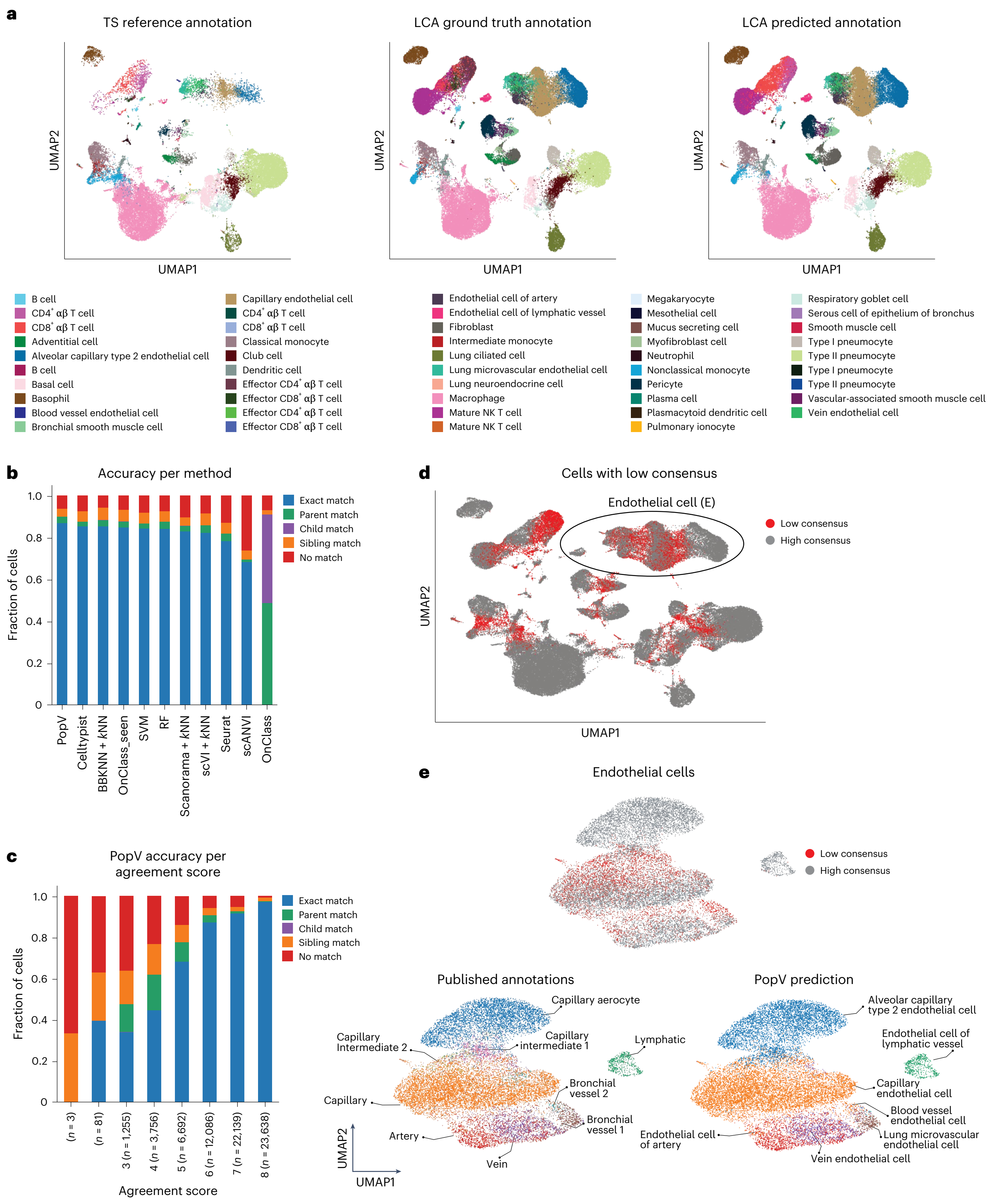


Fig. 2 | PopV prediction on LCA and TS lung as reference is accurate and interpretable. a, UMAP embedding after scANVI integration of TS reference cells, LCA query cells labeled with the ground-truth label and LCA query cells labeled with predicted label. **b**, Ontology accuracy (Methods) for the various methods computed on the query cells. **c**, Ontology accuracy for the prediction scores in popV. **d**, Highlighted cells with a consensus score of 4 or less (low consensus). **e**, Zoomed-in view of endothelial cells in the LCA with popV-predicted labels and ground-truth labels displayed. The zoomed-in picture is rotated by 90° to allow readability of all labels. Alveolar capillary type 2 endothelial cell is the Cell Ontology term for capillary aerocytes. The LCA annotated additional cell types between capillary aerocytes and capillary endothelial cells. TS, Tabula Sapiens; LCA, Lung Cell Atlas.

originating from the Tabula Sapiens reference were labeled as a mixture of CD4 and CD8 T cells. Consequently, most algorithms in popV labeled this low consensus group as a mix of CD4 and CD8 T cells with different decision boundaries. Manually following up on this low-scoring group, we checked the marker gene *CD8A* and found a clear decision boundary that distinguishes the CD4$^+$ (T$_H$) and CD8$^+$ (cytotoxic) subsets in a manner consistent with the (hidden) query annotation. Despite this clear delineation, we found that many CD8$^-$ cells are labeled in the reference atlas as CD8$^+$ T cells. A low consensus score in this group of cells helped to identify wrongly annotated cells in the reference dataset, and we highlight that manual scrutiny can clean up these wrong labels.

PopV uses a diverse set of underlying classifiers. The heuristic is that including methods with varying bias allows us to detect uncertain predictions. We studied whether distinct classifiers are essential by comparing popV against majority voting between eight different SVM algorithms with different kernels and cost parameters. We find the popV consensus score to better correlate with accuracy than this simplified algorithm (Extended Data Fig. 4a,b,d). Indeed, we find higher diversity in the predicted cell-type labels in the predictors underlying popV and find no pair of predictors with a Hamming similarity above 0.9 (Extended Data Fig. 4c). While popV highlighted problems with annotating different subsets of T cells, majority voting after SVM shows high uncertainty for cells predicted to be natural killer (NK) cells. However, for these cells, we found marker gene expression that aligns with those cells being NK cells highlighting an accurate prediction by popV. Taking together, using a diverse set of algorithms enables popV to highlight cell types with wrongly predicted labels, while a more simplified algorithm using only predictors based on SVM does not provide a calibrated classifier.

We have demonstrated here that the consensus score can highlight regions that require manual scrutiny and that reannotation in those regions can be performed using marker gene expression. This leads to new delineation of cell types not discovered in the reference dataset, detection of query-specific cell types and correction of the cell type label of wrongly assigned reference cell type labels.

PopV provides useful label transfer in case of drastic differences in cellular composition

After highlighting that popV is capable of detecting query-specific cells and that the consensus score is capable of highlighting these cells, we studied whether this can also be achieved when we have very different query and reference datasets. To this end, we studied the annotation of thymus cells using Tabula Sapiens as a reference dataset and a second study, which profiled thymi from different age groups (fetal, childhood, adolescence and adulthood) as query[22] (Supplementary Fig. 5). In particular, the thymus undergoes involution with age, and the adult thymus, which we use here as reference, does not accurately represent the structure and function of the thymus in younger individuals. In particular, we anticipate that the reference sample will not provide ample representation of the developing T cell population, which is prevalent in our query data.

UMAP embedding of the two harmonized datasets clearly highlights the subsets of query cells that are represented in the reference dataset, while as expected by the age of the donors in the Tabula Sapiens project, the compartments of thymocytes and developing T cells are almost absent from the reference dataset (Fig. 3a and Supplementary Fig. 6). Indeed, we find a high consensus score and accuracy in regions well represented in the reference dataset, while the consensus score for query-specific cell types is lower (Fig. 3b,c). We find a high prediction accuracy for query cells from the adult thymus that are similar to the cells from the reference dataset (Fig. 3d and Supplementary Fig. 7). Reassuringly, popV assigned low consensus scores to the majority of cells from compartments that are absent in adult humans, highlighting them for manual annotation (Fig. 3e and Extended Data Fig. 5), while for cells isolated from an adult thymus, we overall find very high accuracy across all algorithms and a high consensus score with popV outperforming the single methods in accuracy (Fig. 3f and Supplementary Fig. 8). We found for the underlying algorithms that their internal certainties do not correlate well with the accuracy and find a lower area under the precision–recall curve for these algorithms compared to popV (Extended Data Fig. 6 and Supplementary Fig. 9).

We identified two other cell populations that are underrepresented in Tabula Sapiens compared to the query dataset, which are cortical thymic epithelial cells (also associated with involution[23]) and plasmacytoid dendritic cells. Similarly to our previous examples, we find that the consensus score associated with cortical epithelial cells is indeed low, with a variety of annotations assigned to these cells by the different algorithms, including fibroblasts and medullary epithelial cells (Supplementary Fig. 10). The low consensus score suggests that manual curation of this group of cells is needed. In this case, the manual assignment of the correct out-of-reference label is relatively straightforward using *PSMB11*, an established marker of cortical thymic epithelial cells that is not expressed in any cell-type in the Tabula Sapiens reference.

For plasmacytoid dendritic cells, all algorithms except Scanorama + *k*NN predicted that those cells are B cells or plasma cells. Scanorama+ *k*NN predicted that those cells are dendritic cells. Even OnClass, which can predict cells not present in the reference dataset, predicted those cells as antibody-secreting cells or lymphocytes of B lineage, with not a single cell correctly predicted as a plasmacytoid dendritic cell. However, these query cells expressed high levels of *CLEC4C* and *IL3RA* and were therefore correctly labeled as plasmacytoid dendritic cells. As two-thirds of plasmacytoid dendritic cells have a score of 5 or lower, manual identification of these cells is possible, and the user can identify those confidently wrongly annotated cells using these marker genes.

The only cell fraction that had a high consensus score but low accuracy is a group of cells labeled as endothelial cells by popV, while annotated as lymphocytes in the original (hidden) annotation of the query dataset. However, these cells express *CAVIN2*, *TFPI*, which fits well with an annotation as endothelial cells. We found that their gene expression aligns well with lymphatic endothelial cells. Therefore, it suggests a wrong annotation in the query dataset and a correct prediction by popV.

Overall, this demonstrates that the consensus score yields an interpretable metric for prediction accuracy and that it helps handle cases of discrepancies between the query and reference dataset.

Creating consensus cell-type labels across different sequencing technologies is important with the advent of large integration efforts. We applied popV to this scenario by transferring labels from the Lung Cell Atlas (Fig. 2) to two lung datasets, one sequenced using Drop-seq technology and the other using nuclei sequencing[24,25]. We find an accurate transfer of labels for both datasets (Extended Data Fig. 7). The popV consensus score again provides a metric that was better correlated with accuracy compared to the intrinsic certainty metrics of the underlying algorithms (Supplementary Figs. 11–14). Thus, it allows detecting cells that require manual scrutiny in these more challenging scenarios.

PopV relies on using a Cell Ontology to aggregate votes across the ontology. For some reference datasets, cells are not annotated according to an existing Cell Ontology. We therefore tested the impact when no matching ontology is provided. For this case, we disable the use of OnClass, which requires an ontology (Supplementary Note and Extended Data Fig. 8). We find popV to perform well in this scenario. Additionally, this dataset allowed us to study the effect of different cell-type granularities on cell-type prediction. The prediction score highlights high granularity annotations as less confidently transferred and thereby requiring more manual scrutiny to perform annotation at this very fine level.

Finally, we performed ablation studies to test whether algorithms can be removed from popV while maintaining accurate and well-interpretable results (Extended Data Fig. 9). After dropping three

https://doi.org/10.1038/s41588-024-01993-3

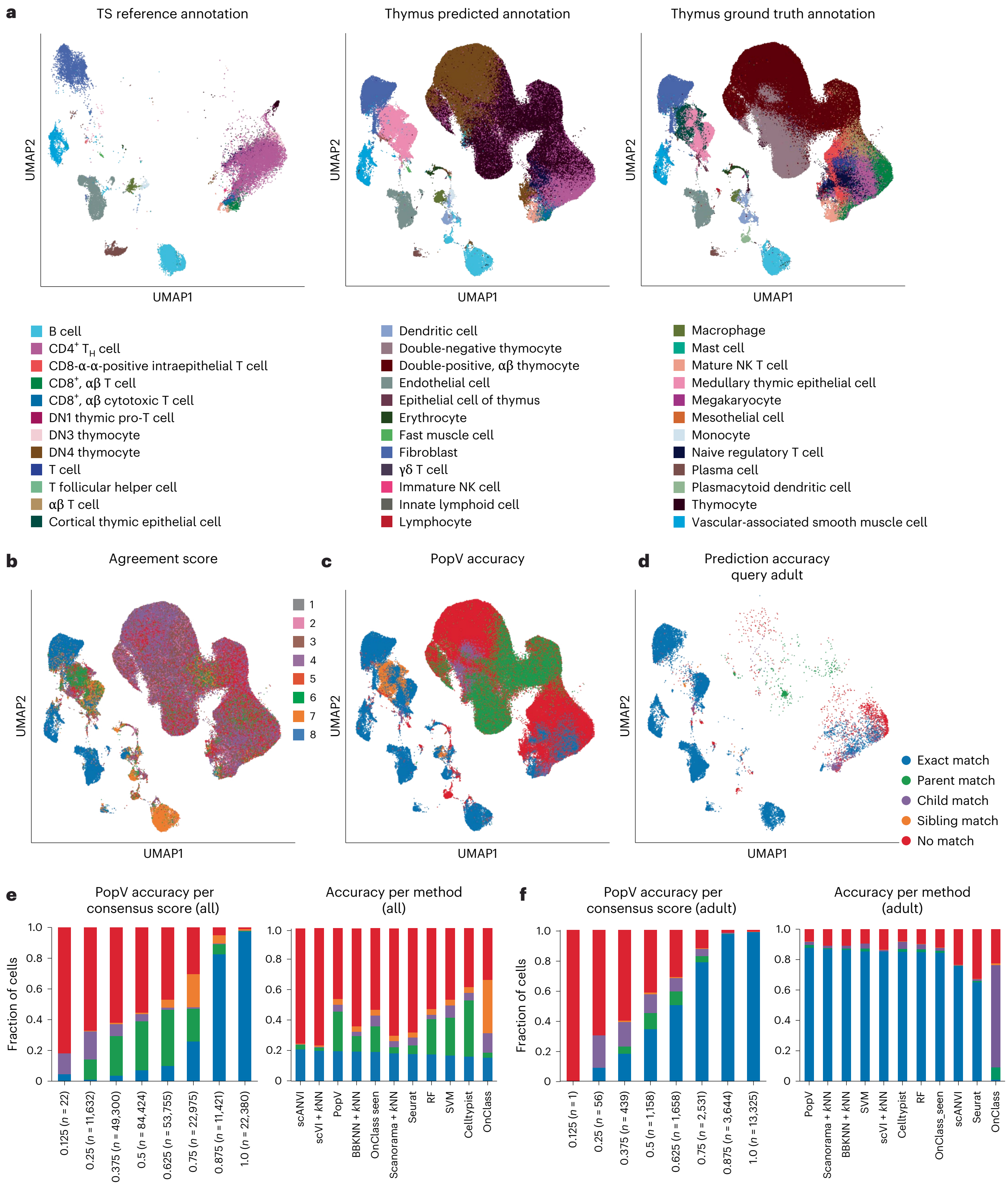


Fig. 3 | PopV identifies thymocytes as query-specific cell types and yields highly interpretable consensus scores. a, UMAP embedding after scANVI integration of reference cells (TS) and query cells (thymus cells across different age groups) labeled by popV prediction and original annotation. **b**, PopV prediction score overlaid on the UMAP plot. The prediction score is low for thymocytes and higher for most other cell types. **c**, The prediction accuracy of the popV prediction highlights the low accuracy in developing thymocytes. **d**, The prediction accuracy of the popV prediction in adult thymus cells in the query shows high accuracy except for CD8 T cells. **e**, Left, PopV accuracy and consensus score are well correlated in all thymus cells with high accuracy for predictions with a consensus score of 7 and 8. Right, All methods show a low accuracy on fetal cells. **f**, Left, PopV accuracy and consensus score are also well correlated when subsetting to cells from adult donors. Right, PopV shows the highest accuracy when subsetting to adult cells; most methods show similarly high accuracy.

of the eight algorithms, we found that the accuracy remains stable in the tested cases. However, we find a decrease in the ability to distinguish predictions with low accuracy. We, therefore, conclude that the number of algorithms is essential to be able to highlight low-confidence predictions.

Discussion

We have developed popV, an ensemble method for cell type annotation, to yield an interpretable certainty quantification for the task of cell type annotation. We have demonstrated throughout this manuscript that in various scenarios with different sequencing technologies, various cell type resolutions and various overlaps of reference and query datasets, popV yields a confidence score that is well correlated with the actual accuracy of cell type transfer. We demonstrated that the prediction score can predict cell types that are specific to the query dataset (middle temporal gyrus (MTG)-specific neurons), incorrectly annotated in the reference (CD4 T cell subsets in Tabula Sapiens) or in the query dataset (lymphatic endothelial cells in the thymus) or cell types that are not annotated in the reference dataset while present in both datasets (lung intermediate capillary endothelial cells in Tabula Sapiens).

PopV is implemented as an easy-to-install, open-source Python tool. The codebase is designed so that adding additional cell type classification algorithms is straightforward, thereby allowing researchers to mitigate the risk of choosing a single algorithm (that is, circumvent the no 'one size fits all' problem). We expect future annotation tools to be developed and popV to be used as a tool to handle various biases in these tools and to help quantify certainty in automatic prediction. As an example, upon user request, we included Harmony + *k*NN, which was not part of the initial release and therefore not used throughout the manuscript, as a classification model and found popV's flexible framework to be straightforward in implementing new predictors.

PopV's performance is limited by the performance of the underlying predictors. We showed throughout the manuscript that overall popV performed equally well as the single-best method in terms of accuracy. However, the aim of popV is not to improve the accuracy of cell-type annotation over the single predictors but to yield a metric of certainty that is easy to interpret and well-calibrated. In fact, we found that algorithm-intrinsic certainties tend to be poorly correlated with the accuracy of cell-type annotation. While single methods provide similar calibration as popV, no other method provides throughout all case studies similar performance. Other methods therefore provide no reliable calibration. This is also reflected in a recent study that highlights the low calibration of conventional tools for cell-type transfer[26]. Conversely, we demonstrated that the popV consensus score is highly associated with accuracy and that it helps identify cases where manual involvement is required. A future focus in the development of cell-type annotation tools will be on providing better internal certainty measurements. Including these algorithms in popV will provide even more interpretable results for label transfer, as, for example, in the case of several algorithms having low internal certainty but agreeing on the most probable label. Label transfer enables harmonizing cell-type annotations across different datasets and streamlines data analysis. However, we recommend validating the transferred labels by checking the expression of canonical marker genes because cell-type label transfer can fail. For example, for plasmacytoid dendritic cells in the thymus dataset, for which the corresponding cell type was missing from the reference dataset, these cells were incorrectly annotated as B cells.

We included eight different algorithms in popV and developed popV in a way that allows the inclusion of new algorithms in a straightforward manner. We performed ablation studies and found that removing algorithms led to less interpretable consensus scores and made distinguishing low-confidence predictions harder. In the future, these metrics will allow us to scrutinize, including additional algorithms or remove existing ones. Across the case studies here and in applications to other datasets, we find an accuracy greater than 95% for scores of seven and eight out of eight predictors, while we recommend manually scrutinizing predictions with a consensus score of six out of eight and less. We highlighted that the good calibration in popV is dependent on a diverse set of predictors. These predictors have different biases (integration methods with *k*NN classifier versus supervised classifier in gene expression space). PopV is a flexible framework that can easily be extended to new classifiers. To make popV a valuable resource for the community, we provide a Google Colab notebook with pretrained models for every tissue in the final Tabula Sapiens publication.

Online content

Any methods, additional references, Nature Portfolio reporting summaries, source data, extended data, supplementary information, acknowledgements, peer review information; details of author contributions and competing interests; and statements of data and code availability are available at https://doi.org/10.1038/s41588-024-01993-3.

References

1. Luecken, M. D. & Theis, F. J. Current best practices in single-cell RNA-seq analysis: a tutorial. *Mol. Syst. Biol.* **15**, e8746 (2019).
2. Lähnemann, D. et al. Eleven grand challenges in single-cell data science. *Genome Biol.* **21**, 31 (2020).
3. Wagner, A., Regev, A. & Yosef, N. Revealing the vectors of cellular identity with single-cell genomics. *Nat. Biotechnol.* **34**, 1145–1160 (2016).
4. Pasquini, G., Rojo Arias, J. E., Schäfer, P. & Busskamp, V. Automated methods for cell type annotation on scRNA-seq data. *Comput. Struct. Biotechnol. J.* **19**, 961–969 (2021).
5. Clarke, Z. A. et al. Tutorial: guidelines for annotating single-cell transcriptomic maps using automated and manual methods. *Nat. Protoc.* **16**, 2749–2764 (2021).
6. Osumi-Sutherland, D. et al. Cell type ontologies of the Human Cell Atlas. *Nat. Cell Biol.* **23**, 1129–1135 (2021).
7. Jones, R. C. et al. The Tabula Sapiens: a multiple-organ, single-cell transcriptomic atlas of humans. *Science* **376**, eabl4896 (2022).
8. Pedregosa, F. et al. Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011).
9. Xu, C. et al. Probabilistic harmonization and annotation of single-cell transcriptomics data with deep generative models. *Mol. Syst. Biol.* **17**, e9620 (2021).
10. Wang, S. et al. Leveraging the Cell Ontology to classify unseen cell types. *Nat. Commun.* **12**, 5556 (2021).
11. Conde, C. D. et al. Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science* **376**, eabl5197 (2022).
12. Lopez, R., Regier, J., Cole, M. B., Jordan, M. I. & Yosef, N. Deep generative modeling for single-cell transcriptomics. *Nat. Methods* **15**, 1053–1058 (2018).
13. Polanski, K. et al. BBKNN: fast batch alignment of single cell transcriptomes. *Bioinformatics* **36**, 964–965 (2020).
14. Hie, B., Bryson, B. & Berger, B. Efficient integration of heterogeneous single-cell transcriptomes using Scanorama. *Nat. Biotechnol.* **37**, 685–691 (2019).
15. Abdelaal, T. et al. A comparison of automatic cell identification methods for single-cell RNA sequencing data. *Genome Biol.* **20**, 194 (2019).
16. Luecken, M. D. et al. Benchmarking atlas-level data integration in single-cell genomics. *Nat. Methods* **19**, 41–50 (2021).
17. Korsunsky, I. et al. Fast, sensitive and accurate integration of single-cell data with Harmony. *Nat. Methods* **16**, 1289–1296 (2019).
18. Ergen, C. PopV pretrained models for Tabula Sapiens reference and helper files. *Zenodo* https://doi.org/10.5281/zenodo.7580707 (2023).

19. Lotfollahi, M. et al. Mapping single-cell data to reference atlases by transfer learning. *Nat. Biotechnol.* **40**, 121–130 (2021).
20. Travaglini, K. J. et al. A molecular cell atlas of the human lung from single-cell RNA sequencing. *Nature* **587**, 619–625 (2020).
21. Stuart, T. et al. Comprehensive integration of single-cell data. *Cell* **177**, 1888–1902 (2019).
22. Park, J.-E. et al. A cell atlas of human thymic development defines T cell repertoire formation. *Science* **367**, eaay3224 (2020).
23. Rode, I. & Boehm, T. Regenerative capacity of adult cortical thymic epithelial cells. *Proc. Natl Acad. Sci. USA* **109**, 3463–3468 (2012).
24. Wang, A. et al. Single-cell multiomic profiling of human lungs reveals cell-type-specific and age-dynamic control of SARS-CoV2 host genes. *eLife* **9**, e62522 (2020).
25. Strunz, M. et al. Alveolar regeneration through a Krt8^{+} transitional stem cell state that persists in human lung fibrosis. *Nat. Commun.* **11**, 3559 (2020).
26. Engelmann, J. et al. Uncertainty quantification for atlas-level cell type transfer. Preprint at *arXiv* 10.48550/arXiv.2211.03793 (2022).

Methods

PopV is a Python package available via the Python Package Index. Further details on popV, the source code and a model tutorial are available at https://github.com/YosefLab/popV. This research complies with all applicable ethical regulations. No primary data were generated. All software versions were used as described in our reproducibility GitHub repository (https://github.com/YosefLab/popv-reproducibility). Specifically, popV (v0.2.2), AnnData (v0.8.0), BBKNN (v1.5.1), Celltypist (v1.6.0), Harmony-Pytorch (v0.1.7), Onclass (v1.3), Scanorama (v1.7.3), Scanpy (v1.9.3), scikit-learn (v0.24.2) and scvi-tools (v1.0.3) were used.

Datasets

Tabula Sapiens. Tabula Sapiens was used throughout the manuscript as the reference dataset. It was downloaded from CELLxGENE (https://cellxgene.cziscience.com/collections/e5f58829-1a66-40b5-a624-9046778e74f5). The expression data were set to the raw object of the h5ad object, which contains count data for all cells and genes. This yields 483,152 cells and 58,559 genes. We filter every cell type that has less than ten cells in a respective tissue as the *k*NN used in popV cannot predict cells with less than eight examples (15 nearest neighbors by default; Supplementary Table 1). We confirmed that all cell types are present in the recent version of the Cell Ontology downloaded from https://github.com/obophenotype/cell-ontology/tree/v2023-02-19. Furthermore, we validated that the cell type annotation was not donor-dependent in Tabula Sapiens. Tabula Sapiens was annotated on a per-donor basis, and for early donors, cell type labels have different names for the same cell type compared to later donors. To reduce the effect of this inconsistency, we excluded several samples (Supplementary Table 2). Additionally, we found a strong batch effect between some 10× samples. After contacting the original authors, we found that the 10× chemistry was the reason for this and created a new metadata column containing the correct assay. The corrected assay can be accessed through https://doi.org/10.5281/zenodo.7587774. All models were trained seperately for each tissue using a batch covariate of concatenated donor and assay (Supplementary Tables 3 and 4).

Lung Cell Atlas. Data were downloaded from CELLxGENE (https://cellxgene.cziscience.com/collections/5d445965-6f1a-4b68-ba3a-b8f765155d3a). We relabeled the cell types to attain conformity with the Cell Ontology (Supplementary Table 6). Additionally, we filtered all blood samples collected for the construction of the Lung Cell Atlas. We created a concatenated column of sample ID and assay and used this concatenated metadata as the query_batch_key in popV (Supplementary Table 5). Throughout this manuscript, the query dataset label was not used as input to scANVI because the general application of popV is intended for an unlabeled query dataset. The Lung Cell Atlas contains 75,071 cells in total, and 39 unique cell types were used as Cell Ontology labels of 59 unique cell types in the original Lung Cell Atlas.

Brain dataset. Data were downloaded from CELLxGENE (https://cellxgene.cziscience.com/collections/283d65eb-dd53-496d-adb7-7570c7caa443). We downloaded the dissection of cerebral cortex (Cx)–precentral gyrus (PrCG)–primary motor cortex-M1C and the dissection of Cx-middle temporal gyrus (MTG) as the two cortical regions with the largest number of cells (Supplementary Table 7). Original cell type labels were used for this dataset, and we used, respectively, cluster_id and supercluster_term as the cell type key. We removed cells labeled with the supercluster terms splatter as well as miscellaneous, as these likely contain low-quality cells where manual annotation was failing. For all downstream metrics, we removed cell types with less than ten cells in each cell type label, as we found those to be reflective of nuclei from distinct brain regions (medium spiny neuron, hippocampal dentate gyrus, hippocampal CA1–3 and amygdala excitatory). We decided against using labels that conform to the Cell Ontology, as all neurons in the original data set were labeled with the same Cell Ontology term neuron, which does not reflect the heterogeneity of these cells. The cell-type labels termed subcluster_id were the finest level of annotation. However, we found little evidence for these labels in the transcriptome of nuclei and excluded those from the analysis.

Thymus dataset. Data were downloaded from https://cellxgene.cziscience.com/collections/de13e3e2-23b6-40ed-a413-e9e12d7d3910 and were analyzed using the same CELLxGENE access link. We labeled cell types to achieve granularity comparable to the reference dataset (Supplementary Table 9). For subset analysis, fetal samples were filtered to every development stage containing a week number (for example, 4th week) as a substring, and adult samples were filtered to the human early adulthood stage. We use the donor ID and assay as the query_batch_key in popV (Supplementary Table 8). The thymus dataset contains 255,901 cells in total, and 28 unique cell types were used as Cell Ontologies of 31 unique cell types in the original thymus dataset. All cells in this dataset were labeled according to the Cell Ontology. However, we decided to summarize all CD4$^+$, as well as all CD8$^+$ T cells, into a common cell type to make the annotation granularity comparable between reference and query datasets (Supplementary Fig. 7). We additionally summarized all B cells in the query and reference dataset to be annotated as B cells, as the label of B cells in Tabula Sapiens showed strong donor inconsistencies and summarized all endothelial cells to be labeled as endothelial cells to harmonize the granularity of cell type labels.

Nucleus sequencing and Drop-seq dataset. Data were downloaded from the extended set of the Human Lung Cell Atlas (https://cellxgene.cziscience.com/e/9f222629-9e39-47d0-b83f-e08d610c7479.cxg/). For nuclei sequencing, we selected the study_id Sun_2020, and for Drop-seq, we selected the study_id Schiller_2020. We used donor_id as the query_batch_key and used no query_labels_key. The models were trained from scratch using 4,000 highly variable genes computed on the reference cells after subsetting to common genes in the query and reference datasets, respectively. Cell types provided in original_ann_nonharmonized were used as ground truth labels based on the manual labels in the original manuscripts[24,25]. We relabeled the cell types to achieve granularity comparable to the reference dataset (Supplementary Tables 10 and 11).

Model parameters

We use eight different cell-type annotation algorithms, and in the following, we explain our parameters for those annotation algorithms as well as the data preprocessing pipeline that we use for popV. For UMAP embedding, we used scanpy default parameters except for a min_dist of 0.3. For the *k*NN classifier, we use uniform weights and n_neighbors equal to 15 in sklearn.neighbors.KNeighborsClassifier. The classifier is first trained on all reference cell labels and is then applied to all query cells in prediction mode. To increase the performance of this classifier, we use a sklearn pipeline and PyNNDescent for neighbor computation[27]. All default parameters for the underlying methods can be changed using a dictionary method_kwargs upon calling popv.annotate_data.

Several algorithms in popV use a graphics processing unit (GPU) to accelerate training. We highly recommend using popV with a GPU. All experiments highlighted here were performed on a workstation with 128 GB of RAM, an Intel Core i9-12900KF desktop CPU and an NVIDIA 3090 GPU. All case studies performed throughout the manuscript were computed on this workstation, and cell-type prediction took less than an hour per dataset. We have applied popV to a dataset of roughly 2 million cells, which takes less than 12 h. We recommend disabling *k*NN on Scanorama for large datasets, as it does not scale well with a large number of experimental batches[28].

Preprocessing. Every dataset was preprocessed using the Process_Query function in popV. The input parameters of Process_Query are explained in the popV documentation. If using a pretrained model folder, both reference and query datasets are subsets of the same genes. It is validated that both datasets contain raw counts. The cell type labels in the reference dataset are subsampled to 300 labeled cells by default to reduce the runtime of the underlying methods. The intersection of genes between the query dataset and the reference dataset is taken, and both datasets are concatenated. We remove all batches in the query and the reference dataset that contain less than nine cells in total, as otherwise BBKNN is failing and no further filtering was performed. Highly variable genes are computed using seurat_v3 flavor in scanpy[29], and by default, 4,000 genes are selected. Count data are stored, and additionally counts are normalized to 10K counts. The log1p function is applied for methods that require normalized data and are stored in a separate layer. For the computation of principal components, the count data are scaled to unit variance. These principal components are used for Spanorama and BBKNN. All keys used to set up the model are stored in the uns field of the anndata object[30].

PopV has the following three different modes:

1. retrain—it trains all methods from scratch and stores the classifier to reuse them on other datasets. This hugely benefits from a GPU to train the scVI and scANVI algorithms as well as the OnClass algorithm.
2. inference—it uses pretrained methods to classify query and reference cells; computes a joint UMAP embedding of query and reference cells and by default uses all eight methods; and trains scVI and scANVI models for 20 epochs using scArches query embedding[19].
3. fast—it uses pretrained methods to classify only query cells; computes a UMAP embedding of query cells if enabled; skips Scanorama and BBKNN data integration as those recompute an embedding instead of projecting cells into an existing embedding; and trains scVI and scANVI models for 1 epoch using scArches query embedding.

BBKNN. Batch-balanced *k*NN is a data integration method. To integrate the datasets, BBKNN takes the nearest neighbors from each batch to construct a balanced neighborhood graph. This nearest-neighbor graph can then be used as a batch-corrected graph embedding of the data[13]. The default settings for popV and those used throughout the manuscript are 50 principal components, 8 neighbors_within_batch and the angular metric. We found that the angular metric outperforms a standard Euclidean metric in our use case. We use the implementation of BBKNN in scanpy.external.pp.bbknn. The batch-balanced nearest neighbors are used as a precomputed metric in sklearn.neighbors.KNeighborsClassifier and used as input for UMAP dimensionality reduction.

Spanorama. Scanorama is a data integration method. It searches for the mutual nearest neighbors across datasets and uses panoramic stitching. Cells are then integrated in PCA space using those mutual neighbors. By default in popV and throughout the manuscript, 50 principal components are used. We compute a new joint embedding of the query and the reference dataset using scanorama.integrate_scanpy function. This joint embedding is used for the *k*NN classification and UMAP embedding.

scVI. ScVI is a variational auto-encoder that incorporates batch keys as latent variables and provides data integration in its latent space. We use the following nondefault parameters for scVI: dropout_rate = 0.05, n_layers = 3, n_latent = 20, gene_likelihood = nb, encode_covariates = True and use_layer_norm = both. The reason for these nonstandard parameters is to facilitate the integration of a query dataset using scArches. For the training parameters, we use by default scVI with n_epochs_kl_warmup = 20 epochs. We compute the joint latent representation of query and reference data, and this joint embedding is used for the *k*NN classification and UMAP embedding.

scANVI. In addition to scVI, a classifier is trained during the training of the auto-encoder on the positions in latent space to classify cells into the provided reference cell type labels. We continue training based on the trained scVI model to reduce the overall training time. For the classifier in scANVI, we use n_layers = 3 and dropout_rate = 0.1. Subsampled labels are used as discussed above. We use as training parameters batch_size = 512 and n_samples_per_label = 20 to stabilize the training of the classifier. Subsequently, the built-in classifier is used to predict cell type labels in the query dataset.

RF. RF uses an ensemble of classification trees together with random feature subsetting to regularize the classification trees. The final prediction is the majority vote across the tree ensemble. We use normalized counts (see above) as input for RF and sklearn.ensemble.RandomForestClassifier as the classifier. We use nondefault parameters as max_features = 200 and class_weight = balanced_subsample as we found the best performance using this parameter combination. For training the classifier, subsampled cell type labels are used as described above, as this improves prediction speed.

SVM. SVMs find the hyperplane that best separates the data. We use sklearn.svm.LinearSVC as the classifier. We use nondefault parameters as $C = 1$, max_iter = 5,000 and class_weight = balanced as we found the best performance using this combination of parameters. For training the classifier, subsampled cell type labels are used as described above for RF, as this improves prediction speed. To allow computation of prediction probabilities, we use sklearn.calibration.CalibratedClassifierCV.

Celltypist. Celltypist uses a logistic regression framework. We use nondefault parameters check_expression = False and max_iter = 500 to allow for faster model training. During celltypist.annotate, we use majority_voting = True except in fast mode where we set it to False. As intrinsic probabilities, we use predictions.probability_matrix as the majority voting purity and not the initial logistic regression probabilities. This is similar to the probabilities used in the Celltypist tutorials.

OnClass. OnClass first computes an embedding of the Cell Ontology using natural language processing (NLP) on the cell type names and then applies random walks. This can be embedded using singular value decomposition[31]. Then a bipartite neural network was optimized to allow classification of the reference cells. The network is then applied to unannotated cells. By design, this allows the classification of unseen cell types in the Cell Ontology term-based low-dimensional embedding. We downloaded the Open Biological and Biomedical Ontology Foundry (OBO) Ontology files in version releases/2023-01-09. To allow fast retraining of sentence embedding, we use sentencetransformer.SentenceTransformer('all-mpnet-base-v2') as the NLP model. This is a newer NLP model than in the original OnClass publication but allows in our hands for faster convergence. We encode all descriptions or cell-type labels in the OBO file. We provide notebooks for retraining with newer releases of ontology files or different species. We provide several ontology files in our GitHub repository; these are cl.obo, which is the downloaded file from https://github.com/obophenotype/cell-ontology. Cl.ontology is a file containing only the is_a cell type relationships from the cl.obo file, and cl.ontology.nlp.emb contains the embeddings of the cl.obo file. As count data, we use normalized data (see above) and disable the options to recompute this normalization in OnClass. OnClass provides the option to use batch integration using Scanorama. We disabled this option to not bias the prediction based on the performance of Scanorama. We found no sign of a strong batch effect in cell-type prediction. For OnClass, we provide two different cell-type labels. OnClass_seen is the prediction of all cells limited to the cell types in the reference dataset, while OnClass_prediction contains for each cell type the final output of the OnClass model. OnClass currently only outputs predictions after step 2 for 10% of the cells, although it computes those on all cells. We found this procedure wasteful and implemented our own version that outputs labels after step 2 on all cells.

Harmony. Harmony was not used throughout the manuscript. However, we found it to scale better to large-scale datasets (more than a million cells and 50 batches) than Scanorama. Harmony uses soft *K*-means clustering and shifts the centroids of those *K*-means clusters to allow batch correction[17]. We used 50 principal components as default input to Harmony. We used an efficient GPU-enabled version of Harmony (https://github.com/lilab-bcb/harmony-pytorch). We computed a new joint embedding of the query and the reference dataset using harmony.harmonize function. This joint embedding is used for the *k*NN classification and the UMAP embedding.

Seurat label transfer. Seurat uses canonical component analysis and nearest-neighbor matching to first integrate the reference dataset and then map the query dataset to this integrated dataset[21]. Seurat is not part of popV, as copying a data frame to R can be a time-consuming step for large data. However, we compared the performance of popV with Seurat for the Lung Cell Atlas. We computed 2,000 genes using FindVariableFeatures with vst transformation in Seurat. We used FindIntegrationAnchors using 30 CCA components. Subsequently, we scaled the corrected counts and calculated 30 principal components. Those components were used in FindTransferAnchors, and afterward, TransferData was called to transfer the labels from the reference dataset to the query dataset.

Consensus voting. As described in the manuscript, we tried majority voting as well as Cell Ontology-based aggregation of OnClass results (Extended Data Fig. 1). We found the Cell-Ontology-based aggregation to outperform majority voting (Supplementary Fig. 1). For this voting strategy, we take the majority vote (counting all predictions) for all predictors except OnClass. For OnClass prediction, we use the predicted cell type and, in addition, every cell type along the path from this cell type to the root node of the Cell Ontology graph and increase the score of those ancestors by 1. We take as consensus score the score at each cell type level node and take as popV prediction the cell type node with the highest score. For majority voting, we use the prediction in OnClass_seen and count the predictors who agree on a certain cell type node. The node with the majority of votes is used as the majority-voted cell type label. If there is a tie between two nodes in the number of votes, we use the cell type label that is further down the Cell Ontology tree, meaning the more granular cell type. If there is still disagreement, we use the cell type that is later in the alphabet to have a deterministic mapping and not rely on the order of cell types in the prediction matrix.

Evaluation metrics

All code for creating evaluation plots is available in the popV package as a _reproducibility module. We will discuss those metrics here. For displaying the translations of cell type terms, we use alluvial plots that highlight the corresponding cell types before and after translation to a Cell Ontology conform term.

Accuracy metrics. If no Cell Ontology graph is available, we use F1 metrics to quantify accuracy. The micro-F1 accuracy computes the number of exact matches across the whole dataset and is a global metric, whereas the macro-F1 score computes a per-cell-type accuracy and averages this across all cell types. The macro-F1 accuracy therefore better represents the performance across rare cell types. We found agreement between both metrics in their evaluation of performance, but the macro-F1 accuracy is more sensitive as rare cell types are harder to predict.

If a Cell Ontology is available, we reasoned that the performance of a predictor is preferable if it predicts a closely related cell type. We therefore computed different matching scores. We computed an exact match similar to the F1 score as cell types that are correctly predicted. Parent match means that the predicted term is a node that is one step closer to the root of the Cell Ontology graph, while child match means that the predicted term is one step further away from the root. Sibling match means that the cell type is two steps away from the correct cell and has the same depth as the original cell type in the Cell Ontology tree. We also experimented with more fine-grained metrics quantifying the distance in the Cell Ontology tree between two cell types. However, after manually checking the corresponding cell types, we found that the nearest matches were the correct metric to evaluate classification, as cell types that are further apart tend to be distinct cell-types.

Confusion matrix. We use scikit-learn.metric.confusion_matrix and normalize those entries. We compute these matrices between all algorithms and the ground-truth label but also between the different algorithms and the consensus label.

Differential expression analysis. We use scanpy.tl.rank_genes_groups with default parameters to yield differentially expressed genes and scanpy.pl.rank_genes_groups_dotplot to plot those results.

Precision–recall curves. We are evaluating whether a decision boundary exists to distinguish between inaccurate predictions and accurate predictions. To this end, whether a prediction is an exact match is used as the label, and the prediction certainty is used as the class probability. Precision–recall curves are computed based on these two values. The area under the precision–recall curve is computed using sklearn.metrics.average_precision_score and the F1 score at the decision boundary using sklearn.metrics.f1_score. To calculate the best decision boundary, we compute the point with the highest F1 score. As some probabilities are discrete, like the popV consensus score, we include all cells at the decision boundary as positive cells. We provide the ratio of cells equal to or above the boundary, the accuracy of those cells and the boundary value in the respective plots.

Ablation experiment. We are evaluating whether removing predictors leads to a reduction in accuracy or calibration for popV. We study this for the thymus case study as well as the scRNA to snRNA label transfer case study. We first remove OnClass from the popV algorithms. After removing, simple majority voting and the popV consensus voting yield the same results. Afterward, we remove all pairs of the seven remaining predictors and compute the majority voting result of these five predictors as well as the consensus score. We compute the weighted F1 score and the area under the precision–recall curve for each set of five predictors. In the title of these plots, we present the value in the original popV algorithm, while in each field of the resulting heatmaps, we present the difference in the respective score between the full popV algorithm and the set of five predictors.

Comparison to majority voting of SVM classifiers. We are evaluating whether majority voting after predicting cell types with a set of SVM classifiers is similar in performance to popV. For this comparison, we use the implementation of SVM in cuML and set up a multiclass classifier using scikit-learn's multiclass.OneVsRestClassifier. We train eight different SVM algorithms that differ in their kernel (sigmoid, polynomial, linear and radial basis function) and the respective cost parameters (0.1 and 1.0). We otherwise used the default hyperparameters in cuML. After predicting the labels for every cell in the Lung Cell Atlas case study, we perform majority voting between the predictions of the underlying SVM predictors and subset the resulting dataset to all query cells. We compute the Hamming similarity by computing 1 − HammingDistance and display the results as the upper triangle comparing all pairs of algorithms against each other. All other metrics in Extended Data Fig. 4 are described in other parts of the Methods and are applied here similarly to the results of both algorithms. We do not display the results of OnClass in Extended Data Fig. 4e to reduce the amount of colors in the resulting plot.

Statistics and reproducibility

No statistical method was used to predetermine the sample size. No data were excluded from the analyses except as highlighted in the Methods. The experiments were not randomized, and the investigators were not blinded to allocation during experiments and outcome assessment.

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

Lung Cell Atlas was downloaded from CELLxGENE (https://cellxgene.cziscience.com/collections/5d445965-6f1a-4b68-ba3a-b8f765155d3a). The thymus dataset was downloaded from https://cellxgene.cziscience.com/collections/de13e3e2-23b6-40ed-a413-e9e12d7d3910. The brain dataset was downloaded from CELLxGENE (https://cellxgene.czisci-ence/com/collections/283d65eb-dd53-496d-adb7-7570c7caa443). Human Lung Cell Atlas was downloaded from CELLxGENE (https://cellxgene.cziscience.com/e/9f222629-9e39-47d0-b83f-e08d610c7479.cxg/). A minimal dataset to reproduce the code is available as a tutorial at https://github.com/YosefLab/popV. All datasets were used from the CELLxGENE census accessed on 9 June 2023. After the correction of metadata (Methods), Tabula Sapiens was deposited on Zenodo (https://doi.org/10.5281/zenodo.7587774)[32].

Code availability

The code to reproduce the experiments of this paper is available at https://github.com/YosefLab/popv-reproducibility (ref. 33). The `popV` package can be found on GitHub at https://github.com/YosefLab/popV. Documentation and tutorials can be found at https://github.com/YosefLab/popV (ref. 34).

References

27. Dong, W., Moses, C. & Li, K. Efficient k-nearest neighbor graph construction for generic similarity measures. *Proceedings of the 20th International Conference on World Wide Web* 577–586 (Association for Computing Machinery, 2011).
28. Wells, S. B. et al. Multimodal profiling reveals tissue-directed signatures of human immune cells altered with age. Preprint at *bioRxiv* https://doi.org/10.1101/2024.01.03.573877 (2024).
29. Wolf, F. A., Angerer, P. & Theis, F. J. SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol.* **19**, 15 (2018).
30. Virshup, I., Rybakov, S., Theis, F. J., Angerer, P. & Alexander Wolf, F. anndata: access and store annotated data matrices. *J. Open Source Softw.* **9**, 4371 (2024).
31. Wang, S., Cho, H., Zhai, C., Berger, B. & Peng, J. Exploiting ontology graph for predicting sparsely annotated gene function. *Bioinformatics* **31**, i357–64 (2015).
32. Ergen, C. Tabula Sapiens filtered data (v1.0). *Zenodo* https://doi.org/10.5281/zenodo.7580683 (2023).
33. Ergen, C. YosefLab/popv-reproducibility. *Zenodo* https://doi.org/10.5281/zenodo.12587267 (2024).
34. Ergen, C., Xing, G., Jayasuriya, M. & Kim, M. YosefLab/PopV: Paper version (v0.9). *Zenodo* https://doi.org/10.5281/zenodo.12561242 (2024).

Acknowledgements

This publication is part of the Human Cell Atlas (www.humancellatlas.org/publications/). We acknowledge the critical feedback provided by the Human Cell Atlas Cell Annotation Platform team. We thank L. Dorman for the extensive benchmarking of popV. We thank the Graphics Department at Weizmann Institute of Science, especially I. Sher, for helping with the graphics in this manuscript. We acknowledge members of the Yosef Laboratories for general feedback. N.Y. was supported by the Chan Zuckerberg Initiative Essential Open Source Software Cycle 4 grant (EOSS4-0000000121) for scvi-tools. C.E. was supported by the German Research Council (DFG Walter Benjamin Stipendium 448802458). A.S. is a Chan Zuckerberg Biohub Investigator. A.P. and N.Y. received funding through grants 2019-203354, 2020-224249, 2021-237288, 2021-006486 and 2022-316725 from the Chan Zuckerberg Initiative DAF, an advised fund of the Silicon Valley Community Foundation and support from the Chan Zuckerberg Biohub. The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript.

Author contributions

C.E., C.X., G.X. and N.Y. conceptualized the study. G.X. and C.E. conceptualized the statistical model with contributions from C.X., M.J. and A.M. C.E. designed and implemented popV with contributions from G.X. and C.X. N.Y., A.P. and A.S. supervised the work. C.E. and N.Y. wrote the paper.

Competing interests

The authors declare no competing interests.

Additional information

Extended data is available for this paper at https://doi.org/10.1038/s41588-024-01993-3.

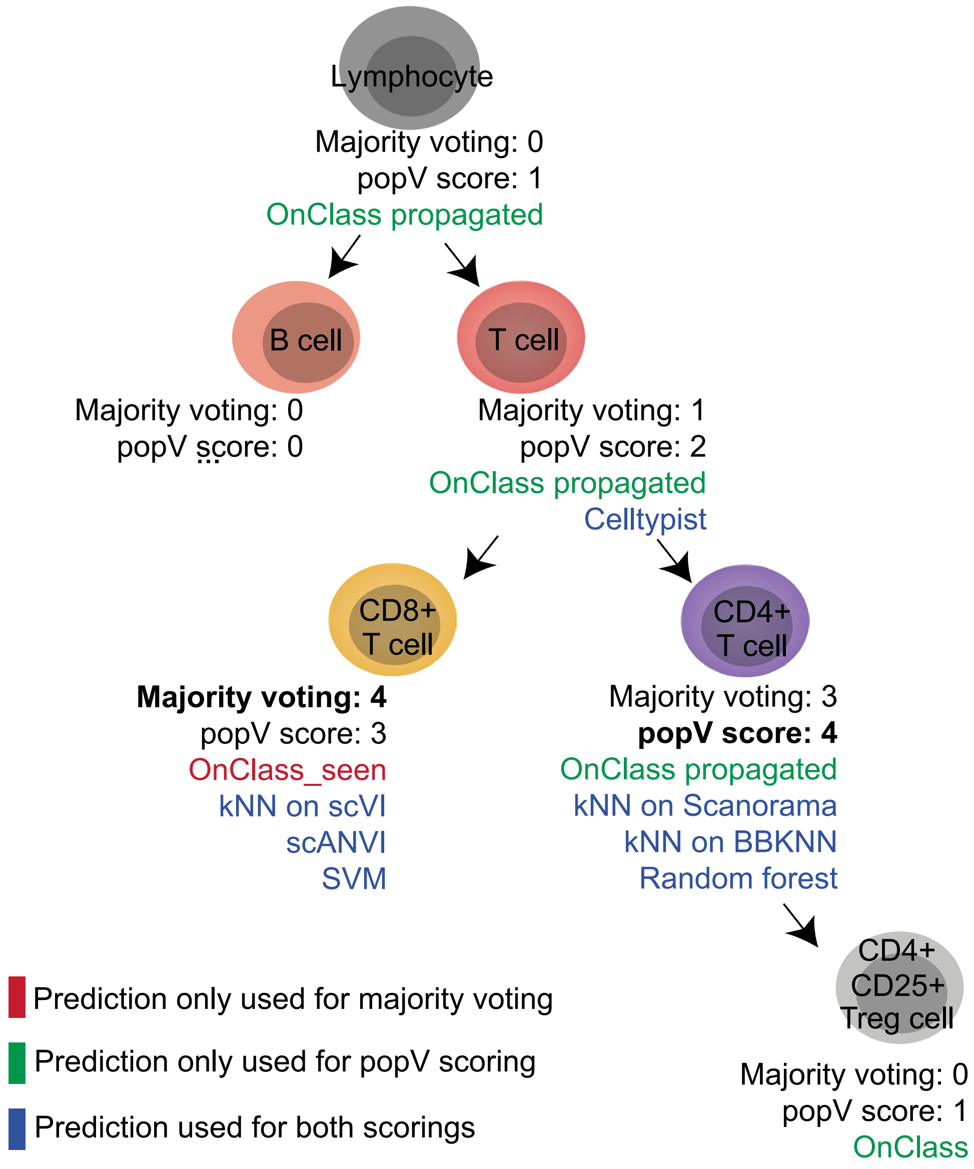
Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41588-024-01993-3.

Correspondence and requests for materials should be addressed to Nir Yosef.

Peer review information *Nature Genetics* thanks Tallulah Andrews, Helene Kretzmer and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

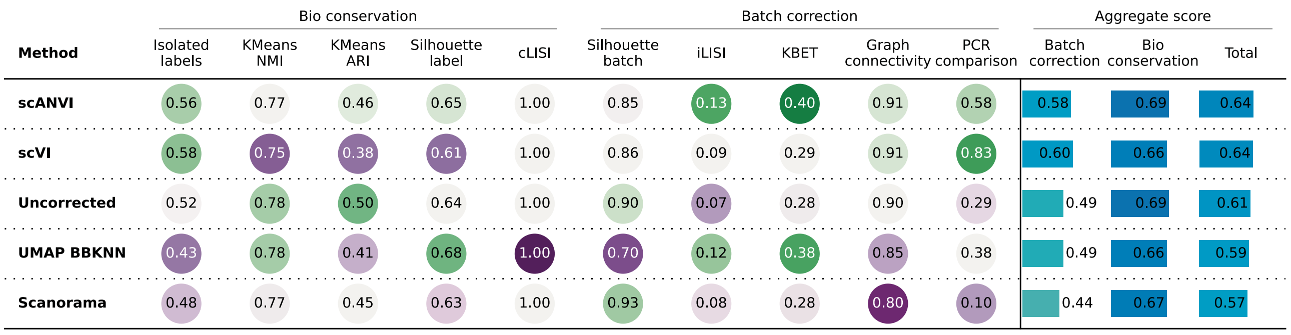
Reprints and permissions information is available at www.nature.com/reprints.

https://doi.org/10.1038/s41588-024-01993-3



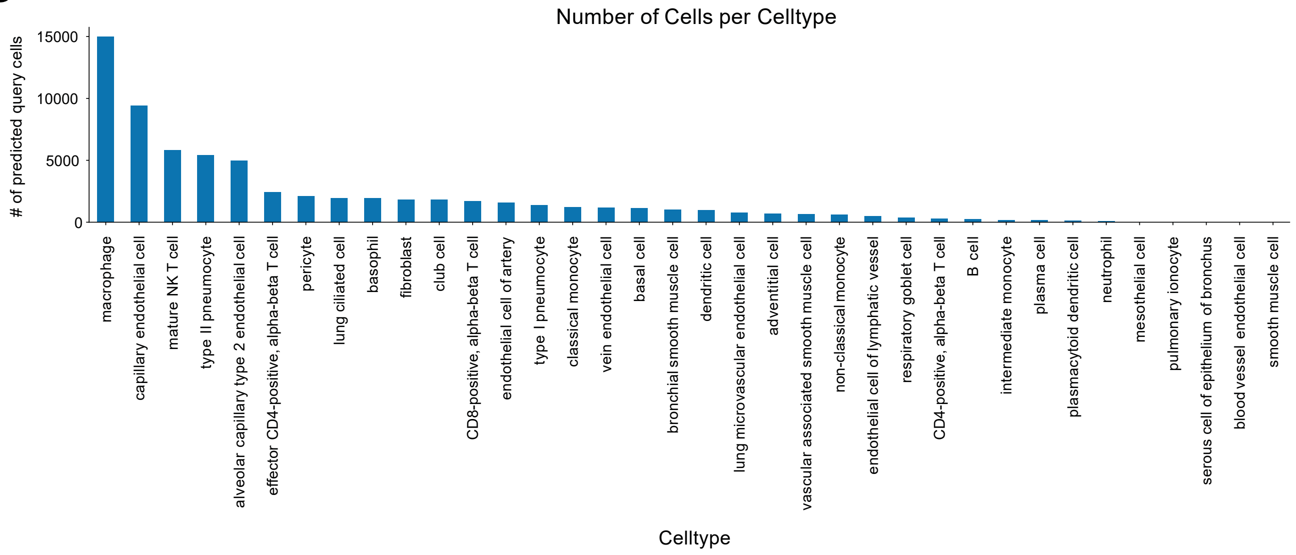
Extended Data Fig. 1 | Comparison of majority voting and popV prediction score. A single cell is annotated by eight different algorithms. OnClass uses a two-step annotation procedure, in which the second can predict cell types that are not part of the reference dataset (here CD4+ CD25$^+$ T$_{reg}$ cell). For simple majority voting, we use the prediction of OnClass at step 1, where it is bound to the cell types observed in the reference dataset (OnClass_seen) and count the predictions of each algorithm. For popV scoring, we propagate the prediction of OnClass along the Cell Ontology graph (shortest path to the root node). Every cell along the path from the root term to the predicted term receives a score of 1, and majority voting is performed for these propagated votes. In the case here, using majority voting, we would classify the cell as a CD8$^+$ T cell with a score of 4, while using the popV consensus score, we would classify the cell as a CD4$^+$ T cell with a score of 4.

A

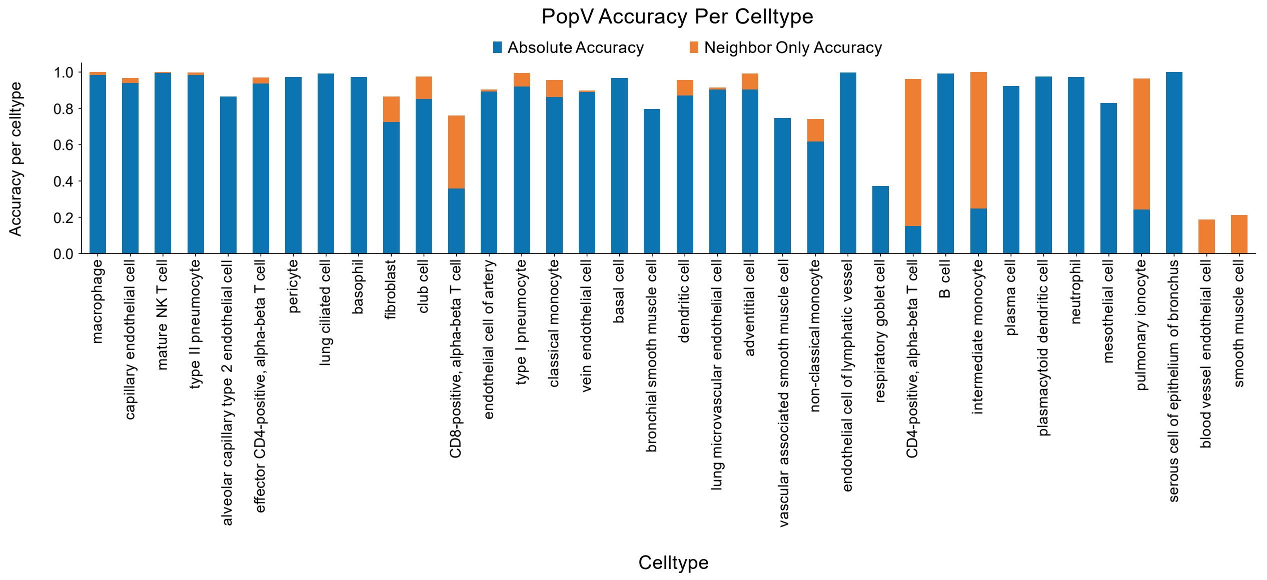


Method	Bio conservation					Batch correction					Aggregate score		
	Isolated labels	KMeans NMI	KMeans ARI	Silhouette label	cLISI	Silhouette batch	iLISI	KBET	Graph connectivity	PCR comparison	Batch correction	Bio conservation	Total
scANVI	0.56	0.77	0.46	0.65	1.00	0.85	0.13	0.40	0.91	0.58	0.58	0.69	0.64
scVI	0.58	0.75	0.38	0.61	1.00	0.86	0.09	0.29	0.91	0.83	0.60	0.66	0.64
Uncorrected	0.52	0.78	0.50	0.64	1.00	0.90	0.07	0.28	0.90	0.29	0.49	0.69	0.61
UMAP BBKNN	0.43	0.78	0.41	0.68	1.00	0.70	0.12	0.38	0.85	0.38	0.49	0.66	0.59
Scanorama	0.48	0.77	0.45	0.63	1.00	0.93	0.08	0.28	0.80	0.10	0.44	0.67	0.57

B

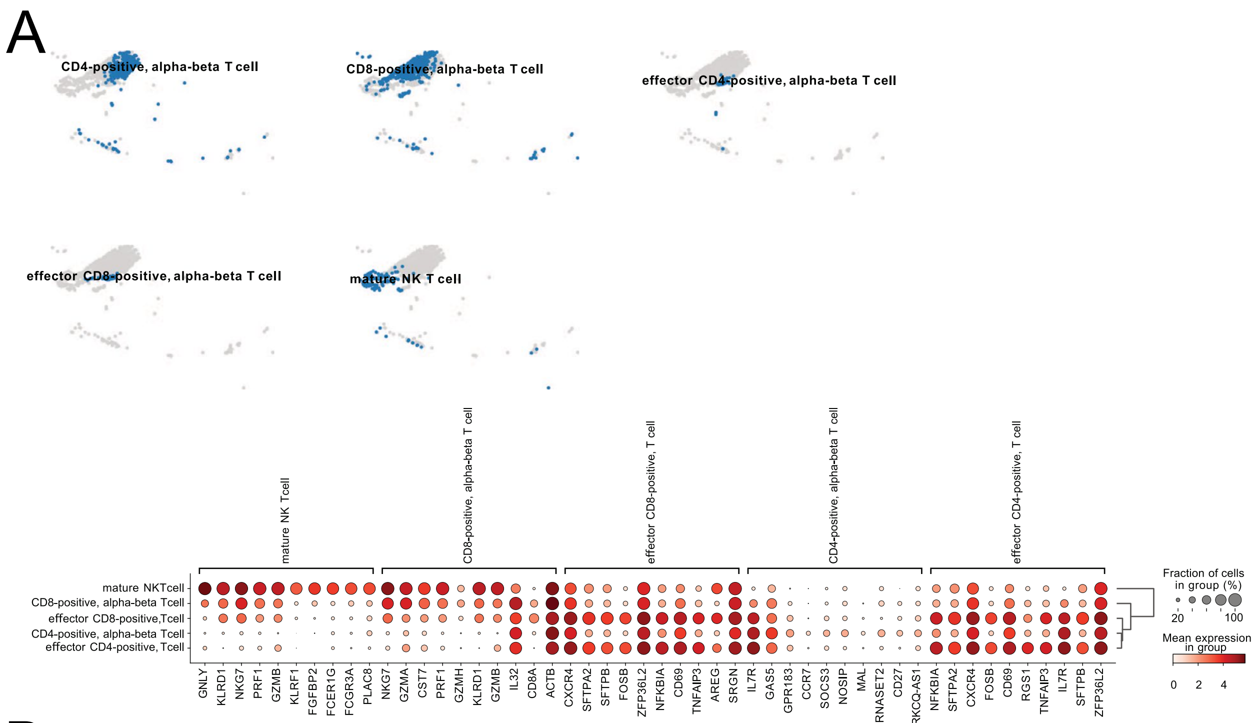


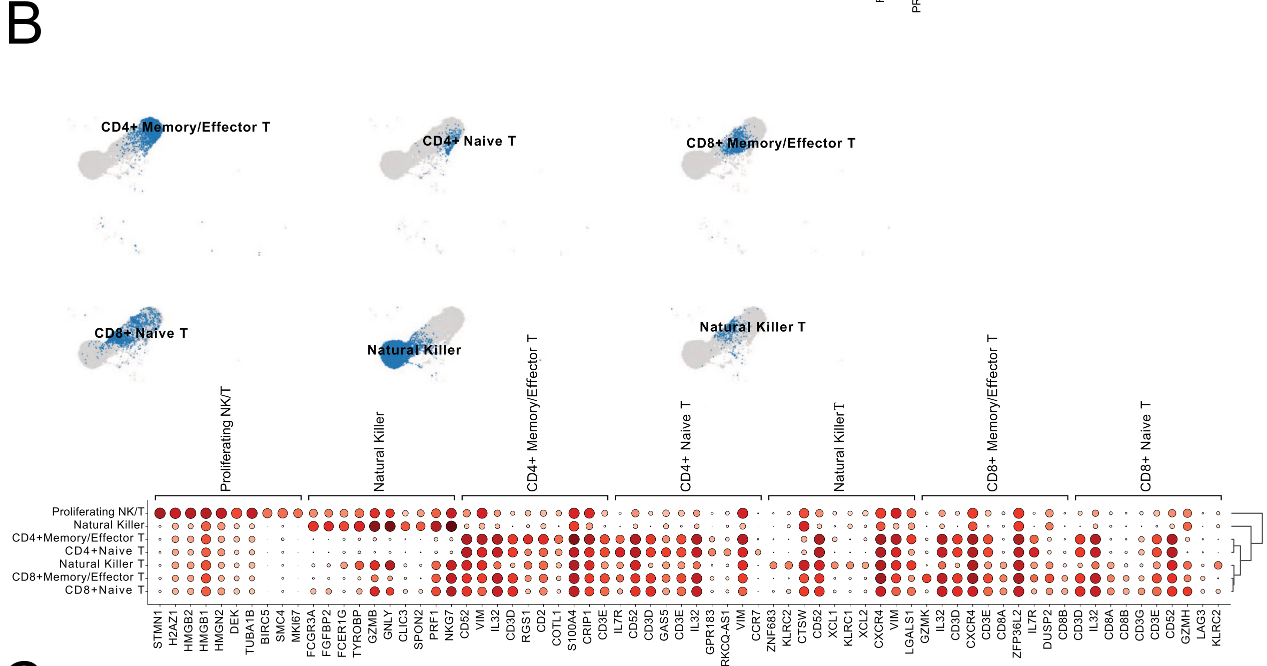
C

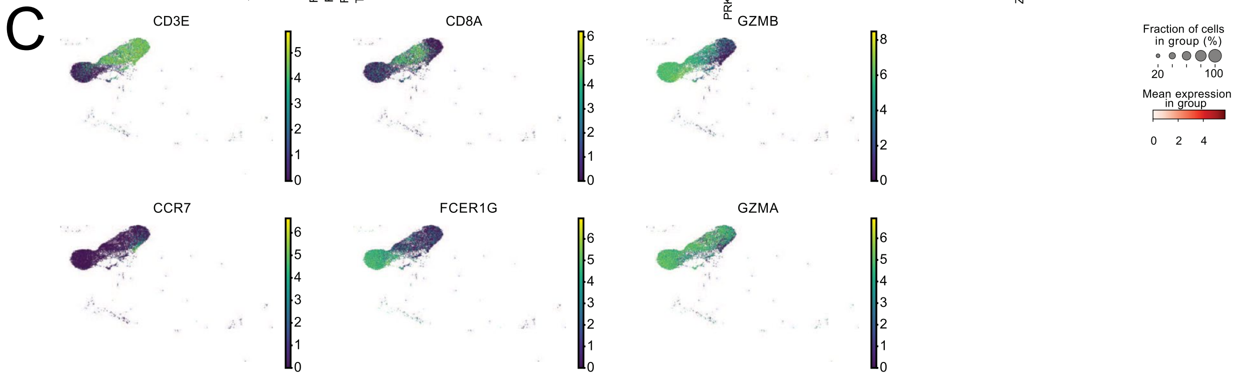


Extended Data Fig. 2 | See next page for caption.

Extended Data Fig. 2 | scANVI shows the highest integration of query cells, and popV shows low confidence for lowly abundant cell types. a, ScIB metrics comparing integration scores after integrating query and reference dataset showed the best integration using scANVI and improvement over uncorrected data. Labels from the original Lung Cell Atlas paper were used to compute cell type-dependent scores, and scores were computed only on query cells. **b**, Displayed is the number of each predicted cell type in query cells and the accuracy for each annotated cell type. **c**, Absolute accuracy corresponds to exact match, while neighbor-only accuracy corresponds to all adjacent cell types (all accuracy terms except no match). Cells that were rarely predicted (smooth muscle cells and blood vessel endothelial cells showed the lowest accuracy). Most cell types have an accuracy greater than 0.9.

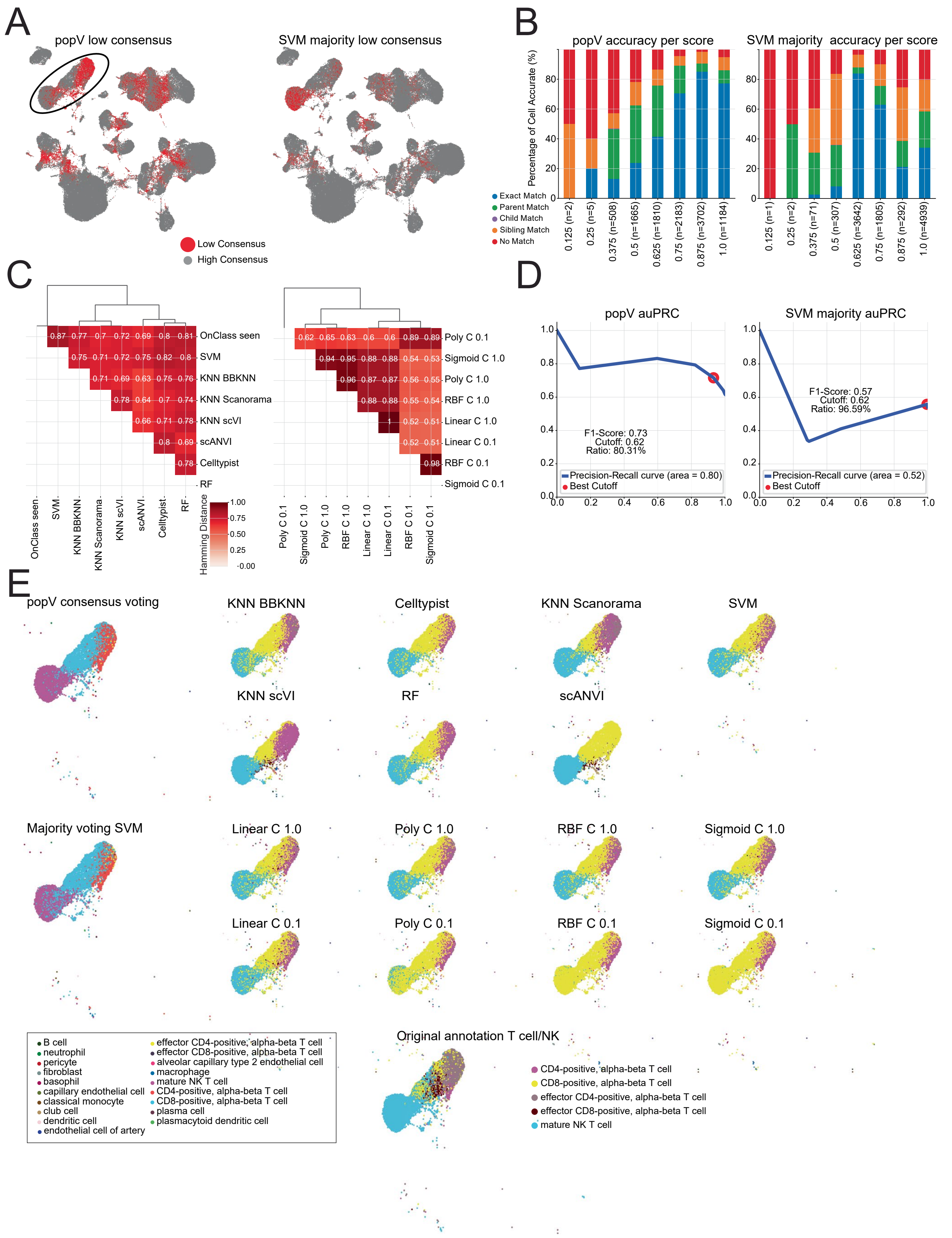






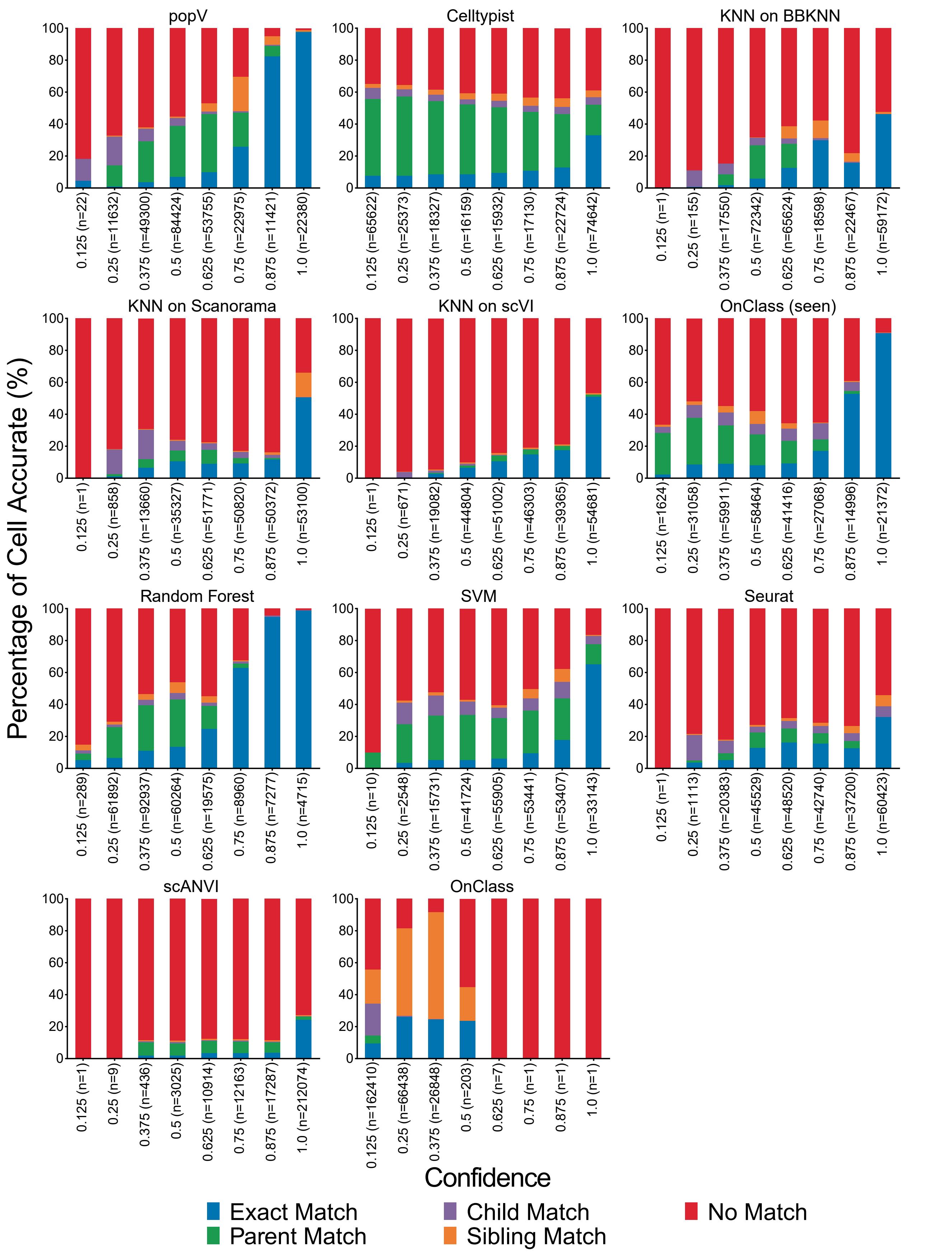
Extended Data Fig. 3 | See next page for caption.

Extended Data Fig. 3 | Analysis of T cell sub-clustering in Lung Cell Atlas and Tabula Sapiens. a, UMAP of cells from the reference dataset labeled with Tabula Sapiens cell type labels highlights the overlap of these labels in integrated space without a clear distinction between CD4 and CD8 T cells. Differential expression analysis identifies surfactant protein genes as markers for annotated effector T cells, which is likely due to ambient counts and no strong marker gene expression in CD4 T cells. **b**, UMAP of cells from the query dataset labeled with cell type labels from the Lung Cell Atlas shows a clear distinction between different cell types. Differentially expressed genes for those cell types align well with the respective literature. **c**, Canonical marker genes for various subtypes show a clear split between T cells and NK cells, as well as CD8 and CD4 T cells. GZMA but not GZMB is also expressed in CD4 T cells.



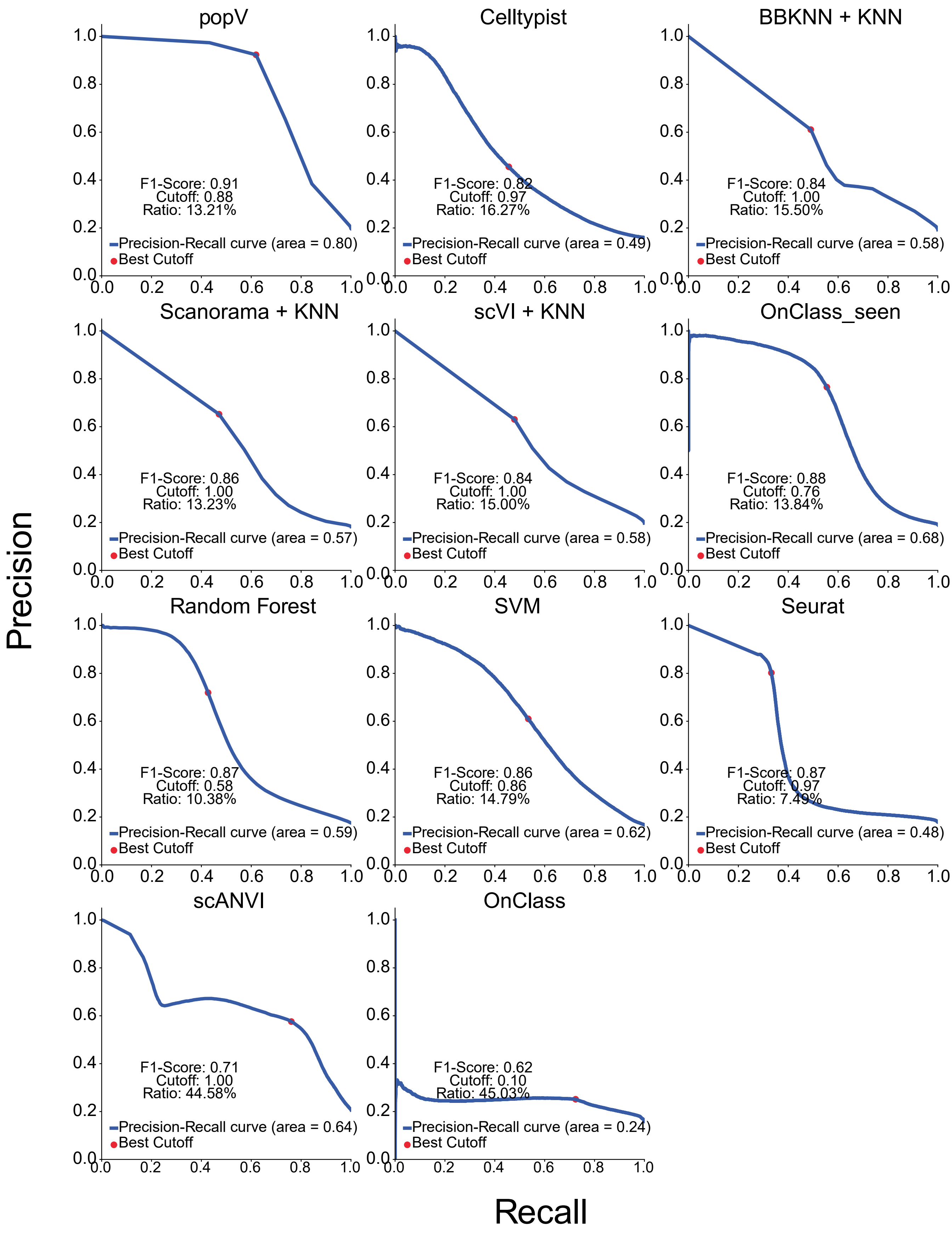
Extended Data Fig. 4 | See next page for caption.

Extended Data Fig. 4 | Comparison of popV against majority voting of multiple SVM classifiers. All plots are displayed for query cells of the lung cell query dataset. We compare popV against majority voting after SVM, for which different kernels (radial basis function, polynomial, linear and sigmoid) and the cost were varied (0.1 and 1.0). **a**, UMAP-highlighting cells with a consensus score of 4 or less (low consensus). We compare popV here with the majority vote of eight distinct SVM classifiers, which differ in their choice of the kernel and **c** parameters. **b**, Accuracy versus consensus score (left) and majority voting score (right) for both consensus algorithms colored by ontology accuracy terms (Methods). **c**, Hamming distance between the underlying classifiers of popV and SVM majority voting. **d**, Precision–recall curves for both algorithms. We used all exact match as the label for the metric calculation. In the legend, area under the precision–recall curve is given (AUPRC). In addition, the F1 score at best decision boundary is given, along with the confidence level at the decision boundary as well as the ratio of cells annotated above this threshold. popV shows the higher AUPRC and F1 scores at the decision boundary. **e**, Focus on T cells and NK cells (circle in **a**). UMAP of the consensus voting as well as underlying classifiers. Top, PopV classifiers are displayed. The main disagreement is between CD8 and CD4 T cells. Middle, Majority voting after SVM shows disagreement between NK and CD8$^+$ T cells. Bottom right, Original annotation of query cells shows agreement for NK cells and no clear separation between different T cell labels.



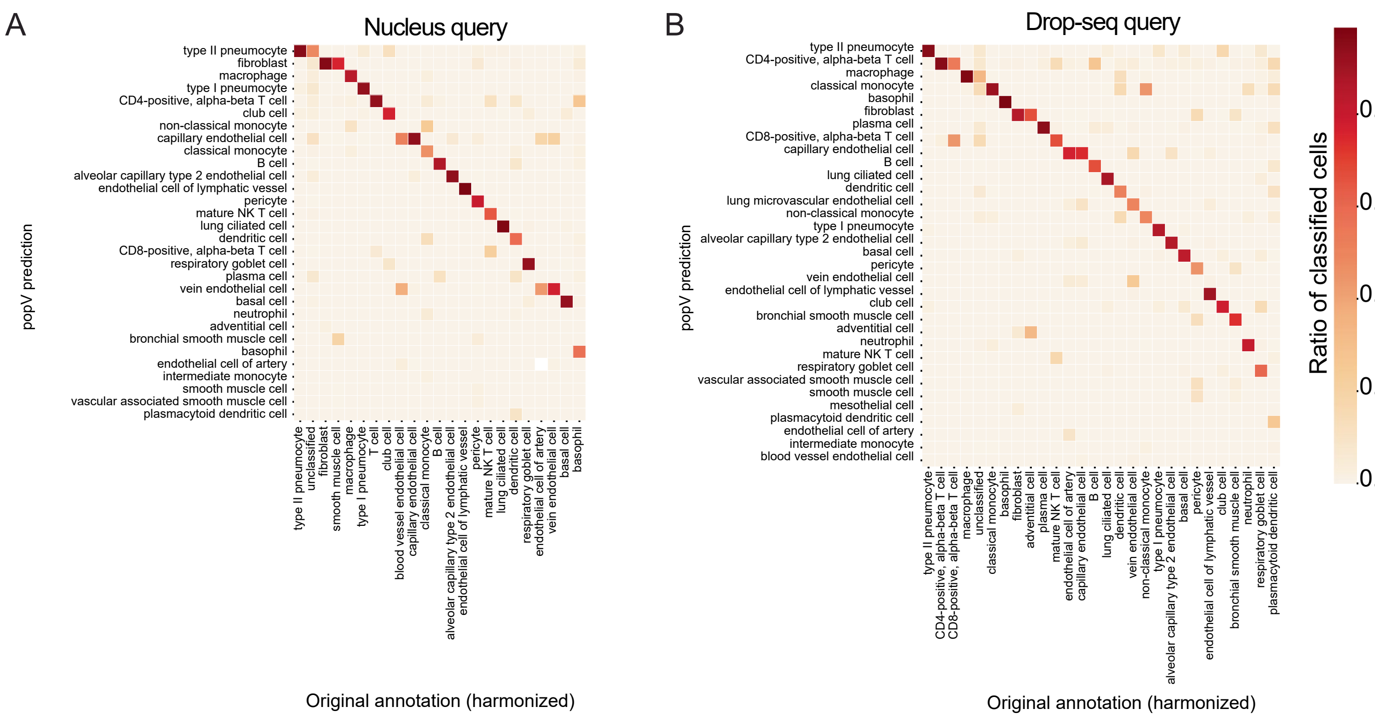
Extended Data Fig. 5 | See next page for caption.

Extended Data Fig. 5 | Calibration of certainty and accuracy for all methods evaluated on all query cells from the thymus. Displayed is the accuracy for bins of internal classification certainty. X-axis labels highlight the number of cells in each bin. PopV, OnClass_seen and random forest show the strongest correlation between exact match and certainty. PopV shows the highest number of high-confidence predictions. Several methods show a high number of incorrect results for predictions with confidence above >87.5%. The coloring of the bars is calculated for the prediction of the respective algorithm, standardized to a height of one per confidence level.

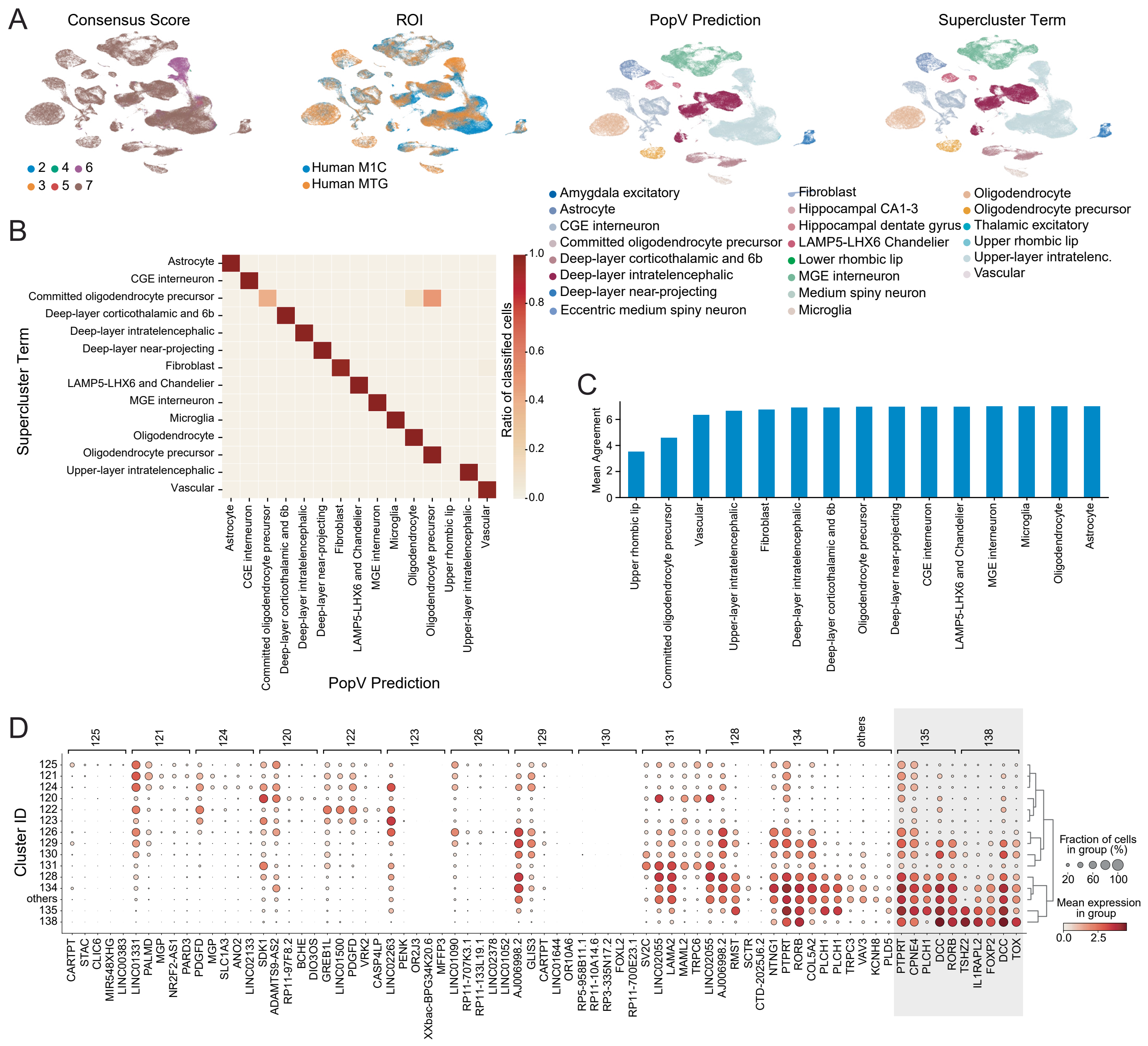


Extended Data Fig. 6 | See next page for caption.

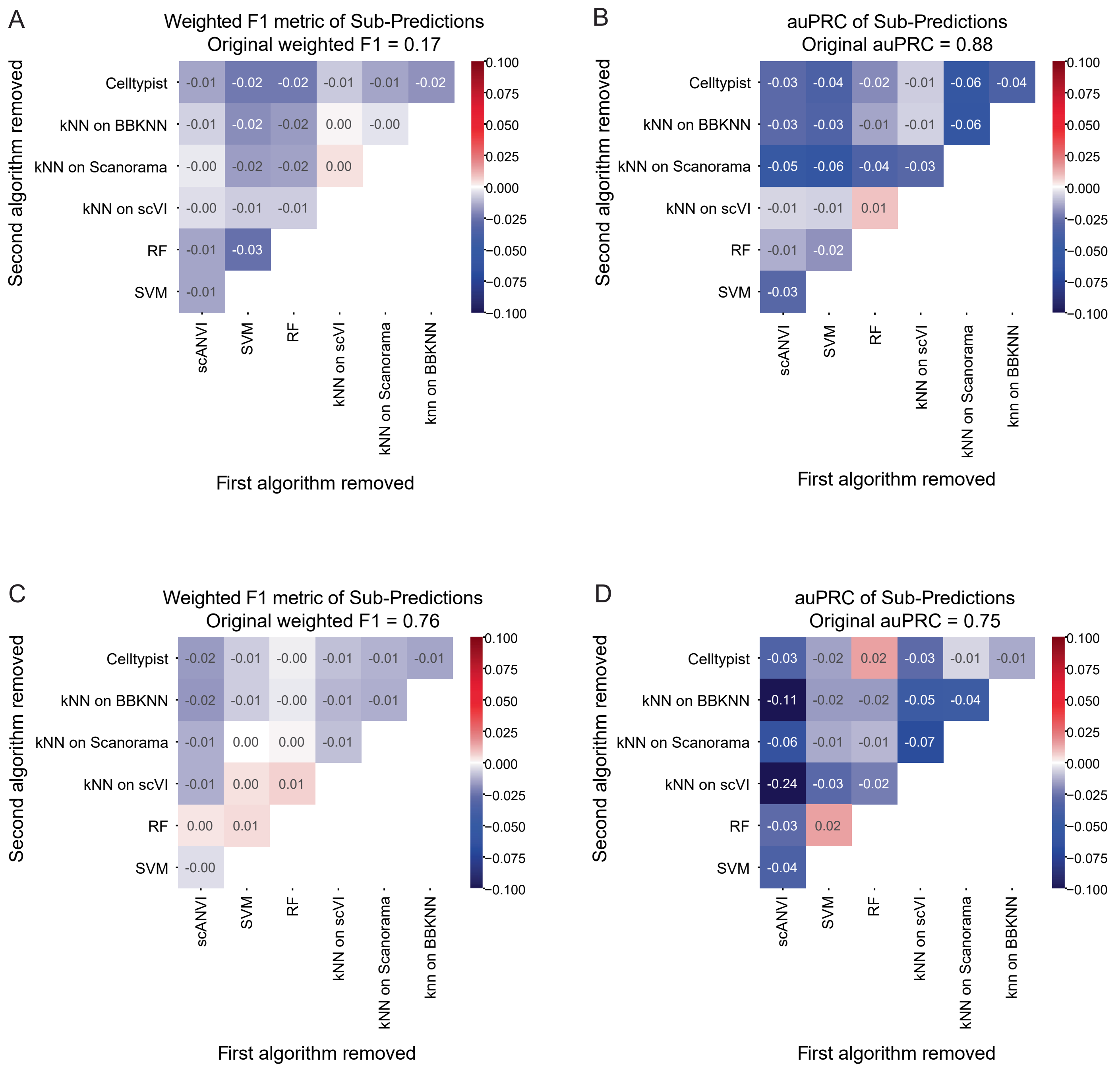
Extended Data Fig. 6 | Precision–recall curves for all classifiers evaluated on all query cells from the thymus. Precision–recall curves are evaluated for all query cells. We used all exact match as the label for the metric calculation. In the legend, area under the precision–recall curve is given (AUPRC). F1 score at best decision boundary is given, along with the confidence level at the decision boundary as well as the ratio of cells annotated above this threshold. PopV shows the highest AUPRC and F1 score at the decision boundary. The decision boundary corresponds to a consensus score of 7 and above.



Extended Data Fig. 7 | Predictions of cell-type labels across different technologies. Confusion matrix of cell-type labels for (**a**) nucleus as well as (**b**) Drop-seq query cells using cells sequenced with 10× from Fig. 2 as the reference dataset. Matrix is column normalized on the respective y-axis; the predicted label from popV is given, and on the x-axis, the original label is provided. Displayed is the ratio of predicted cell types with popV for each ground truth cell type.



Extended Data Fig. 8 | PopV accurately performs labeling of cell types across different brain regions and highlights region-specific neurons. a, UMAP embedding after scANVI integration of reference nuclei (motor cortex, M1G) and query nuclei (medial temporal gyrus, MTG) labeled with consensus score, brain region (ROI), popV prediction and original annotation (supercluster term). **b**, Confusion matrix for the cell types predicted by popV and their respective manual annotations highlights the agreement between both annotations. Displayed is the ratio of predicted cell types with popV for each ground truth cell type. **c**, Mean agreement score per cell type shows that confused cell types also exhibit a lower agreement score and can be detected based on their score. **d**, Differentially expressed genes for cluster ID for upper-layer intratelencephalic neurons. Highlighted are cluster IDs 135 and 138, which are over-represented in the MTG over the M1G. These clusters show an overexpression of FOXP2 and TSHZ2 and are very similar to each other.



Extended Data Fig. 9 | Precision–recall curves for popV prediction when removing classifiers from voting scheme. We evaluated the performance of popV when disabling a subset of algorithms. We removed OnClass from the algorithms so that majority voting and popV consensus scoring yield the same result. We evaluated the performance of popV using a subset of 5 algorithms each. On the respective y- and x-axes, the two algorithms removed from popV for the respective trial are displayed. The difference between the accuracy or auPRC between the original prediction and the prediction after subsetting the algorithm is displayed. Each metric obtained when using all algorithms in popV is given in the plot title. **a** and **b** denote the thymus experiment (Fig. 3) evaluated on all query cells. **c** and **d** correspond to the nucleus dataset (Extended Data Fig. 7).

nature portfolio

Corresponding author(s):	Can Ergen, Nir Yosef
Last updated by author(s):	Jun 26, 2024

Reporting Summary

Nature Portfolio wishes to improve the reproducibility of the work that we publish. This form provides structure for consistency and transparency in reporting. For further information on Nature Portfolio policies, see our Editorial Policies and the Editorial Policy Checklist.

Statistics

For all statistical analyses, confirm that the following items are present in the figure legend, table legend, main text, or Methods section.

n/a	Confirmed	
☐	☒	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement
☐	☒	A statement on whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	The statistical test(s) used AND whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☒	☐	A description of all covariates tested
☒	☐	A description of any assumptions or corrections, such as tests of normality and adjustment for multiple comparisons
☒	☐	A full description of the statistical parameters including central tendency (e.g. means) or other basic estimates (e.g. regression coefficient) AND variation (e.g. standard deviation) or associated estimates of uncertainty (e.g. confidence intervals)
☒	☐	For null hypothesis testing, the test statistic (e.g. *F*, *t*, *r*) with confidence intervals, effect sizes, degrees of freedom and *P* value noted *Give P values as exact values whenever suitable.*
☐	☒	For Bayesian analysis, information on the choice of priors and Markov chain Monte Carlo settings
☒	☐	For hierarchical and complex designs, identification of the appropriate level for tests and full reporting of outcomes
☒	☐	Estimates of effect sizes (e.g. Cohen's *d*, Pearson's *r*), indicating how they were calculated

Our web collection on statistics for biologists contains articles on many of the points above.

Software and code

Policy information about availability of computer code

Data collection

All data used throughout the study was collected from CELLxGENE. For Tabula sapiens, metadata was updated beyond the CELLxGENE version and deposited on Zenodo.

Data analysis

All code used throughout the manuscript is publicly accessible through https://github.com/YosefLab/PopV. For dependencies we refer to the poetry.lock file in the Github repository. Full analysis code and software is available at https://github.com/YosefLab/popv-reproducibility.

```
name: popv_bench
channels:
  - conda-forge
  - defaults
dependencies:
  - _libgcc_mutex=0.1=main
  - _openmp_mutex=5.1=1_gnu
  - asttokens=2.2.1=pyhd8ed1ab_0
  - backcall=0.2.0=pyh9f0ad1d_0
  - backports=1.0=pyhd8ed1ab_3
  - backports.functools_lru_cache=1.6.5=pyhd8ed1ab_0
  - ca-certificates=2023.7.22=hbcca054_0
  - comm=0.1.4=pyhd8ed1ab_0
  - debugpy=1.6.7=py39h6a678d5_0
  - decorator=5.1.1=pyhd8ed1ab_0
  - entrypoints=0.4=pyhd8ed1ab_0
- executing=1.2.0=pyhd8ed1ab_0
- ipykernel=6.25.1=pyh71e2992_0
- ipython=8.14.0=pyh41d4057_0
- jedi=0.19.0=pyhd8ed1ab_0
- jupyter_client=7.3.4=pyhd8ed1ab_0
- jupyter_core=5.3.1=py39hf3d152e_0
- ld_impl_linux-64=2.38=h1181459_1
- libffi=3.3=he6710b0_2
- libgcc-ng=11.2.0=h1234567_1
- libgomp=11.2.0=h1234567_1
- libsodium=1.0.18=h36c2ea0_1
- libstdcxx-ng=11.2.0=h1234567_1
- matplotlib-inline=0.1.6=pyhd8ed1ab_0
- ncurses=6.4=h6a678d5_0
- openssl=1.1.1l=h7f98852_0
- packaging=23.1=pyhd8ed1ab_0
- parso=0.8.3=pyhd8ed1ab_0
- pexpect=4.8.0=pyh1a96a4e_2
- pickleshare=0.7.5=py_1003
- platformdirs=3.10.0=pyhd8ed1ab_0
- prompt-toolkit=3.0.39=pyha770c72_0
- prompt_toolkit=3.0.39=hd8ed1ab_0
- ptyprocess=0.7.0=pyhd3deb0d_0
- pure_eval=0.2.2=pyhd8ed1ab_0
- pygments=2.16.1=pyhd8ed1ab_0
- python=3.9.0=hdb3f193_2
- python-dateutil=2.8.2=pyhd8ed1ab_0
- python_abi=3.9=2_cp39
- readline=8.2=h5eee18b_0
- setuptools=68.0.0=py39h06a4308_0
- sqlite=3.41.2=h5eee18b_0
- stack_data=0.6.2=pyhd8ed1ab_0
- tk=8.6.12=h1ccaba5_0
- traitlets=5.9.0=pyhd8ed1ab_0
- typing_extensions=4.7.1=pyha770c72_0
- wcwidth=0.2.6=pyhd8ed1ab_0
- wheel=0.38.4=py39h06a4308_0
- xz=5.4.2=h5eee18b_0
- zeromq=4.3.4=h9c3ff4c_1
- zlib=1.2.13=h5eee18b_0
- pip:
  - absl-py==1.4.0
  - aiobotocore==2.9.0
  - aiohttp==3.8.5
  - aioitertools==0.11.0
  - aiosignal==1.3.1
  - alabaster==0.7.13
  - anndata==0.8.0
  - annoy==1.17.3
  - anyio==3.7.1
  - array-api-compat==1.4
  - arrow==1.2.3
  - astunparse==1.6.3
  - async-timeout==4.0.3
  - attrs==23.1.0
  - babel==2.12.1
  - backoff==2.2.1
  - bbknn==1.5.1
  - beautifulsoup4==4.12.2
  - bleach==6.0.0
  - blessed==1.20.0
  - botocore==1.33.13
  - cachetools==5.3.1
  - celltypist==1.6.0
  - cellxgene-census==1.9.1
  - certifi==2023.7.22
  - cffi==1.16.0
  - charset-normalizer==3.2.0
  - chex==0.1.7
  - click==8.1.6
  - cloudpickle==3.0.0
  - cmake==3.27.2
  - contextlib2==21.6.0
  - contourpy==1.1.0
  - croniter==1.4.1
  - cryptography==42.0.3
- cubinlinker-cu11==0.3.0.post1
- cuda-python==12.4.0
- cudf-cu11==24.4.0
- cudf-cu12==24.4.0
- cugraph-cu11==24.4.0
- cugraph-cu12==24.4.0
- cuml-cu11==24.4.0
- cuml-cu12==24.4.0
- cupy-cuda11x==13.0.0
- cupy-cuda12x==13.0.0
- cycler==0.11.0
- cython==3.0.0
- dask==2024.1.1
- dask-cuda==24.4.0
- dask-cudf-cu11==24.4.0
- dask-cudf-cu12==24.4.0
- dask-expr==0.4.0
- dateutils==0.6.12
- deepdiff==6.3.1
- defusedxml==0.7.1
- distlib==0.3.8
- distributed==2024.1.1
- dm-tree==0.1.8
- docrep==0.3.2
- docutils==0.17.1
- editables==0.5
- et-xmlfile==1.1.0
- etils==1.4.1
- exceptiongroup==1.1.3
- fastapi==0.98.0
- fastjsonschema==2.18.0
- fastrlock==0.8.2
- fbpca==1.0
- filelock==3.12.2
- flatbuffers==23.5.26
- flax==0.7.2
- fonttools==4.42.0
- frozenlist==1.4.0
- fsspec==2023.12.2
- gast==0.4.0
- gdown==4.7.1
- geosketch==1.2
- google-auth==2.22.0
- google-auth-oauthlib==1.0.0
- google-pasta==0.2.0
- grpcio==1.57.0
- h11==0.14.0
- h5py==3.9.0
- harmony-pytorch==0.1.7
- hatch==1.9.3
- hatchling==1.21.1
- httpcore==1.0.3
- httpx==0.26.0
- huggingface-hub==0.11.1
- hyperlink==21.0.0
- idna==3.4
- igraph==0.10.6
- imagesize==1.4.1
- imgkit==1.2.2
- importlib-metadata==7.0.1
- importlib-resources==6.0.1
- inquirer==3.1.3
- intervaltree==3.1.0
- ipywidgets==8.1.0
- itsdangerous==2.1.2
- jaraco-classes==3.3.1
- jax==0.4.10
- jaxlib==0.4.10
- jeepney==0.8.0
- jinja2==3.1.2
- jmespath==1.0.1
- joblib==1.3.2
- jsonschema==4.19.0
- jsonschema-specifications==2023.7.1
- jupyter-client==7.4.9
- jupyterlab-pygments==0.2.2
- jupyterlab-widgets==3.0.8
- keras==2.13.1
- keyring==24.3.0
- kiwisolver==1.4.4
- leidenalg==0.10.1
- libclang==16.0.6
- lightning==2.0.6
- lightning-cloud==0.5.37
- lightning-utilities==0.9.0
- lit==16.0.6
- llvmlite==0.41.1
- locket==1.0.0
- louvain==0.8.1
- markdown==3.3.4
- markdown-it-py==3.0.0
- markupsafe==2.1.3
- matplotlib==3.7.2
- mdurl==0.1.2
- mistune==3.0.1
- ml-collections==0.1.1
- ml-dtypes==0.2.0
- more-itertools==10.2.0
- mpmath==1.3.0
- msgpack==1.0.5
- mudata==0.2.3
- multidict==6.0.4
- multipledispatch==1.0.0
- natsort==8.4.0
- nbclient==0.8.0
- nbconvert==7.7.3
- nbformat==5.9.2
- nbsphinx==0.9.2
- nbsphinx-link==1.3.0
- nest-asyncio==1.5.7
- networkx==3.1
- nltk==3.8.1
- numba==0.58.1
- numpy==1.24.3
- numpyro==0.12.1
- nvidia-cublas-cu11==11.10.3.66
- nvidia-cuda-cupti-cu11==11.7.101
- nvidia-cuda-nvrtc-cu11==11.7.99
- nvidia-cuda-runtime-cu11==11.7.99
- nvidia-cudnn-cu11==8.5.0.96
- nvidia-cufft-cu11==10.9.0.58
- nvidia-curand-cu11==10.2.10.91
- nvidia-cusolver-cu11==11.4.0.1
- nvidia-cusparse-cu11==11.7.4.91
- nvidia-nccl-cu11==2.14.3
- nvidia-nvtx-cu11==11.7.91
- nvtx==0.2.10
- oauthlib==3.2.2
- obonet==1.0.0
- onclass==1.3
- openpyxl==3.1.2
- opt-einsum==3.3.0
- optax==0.1.7
- orbax-checkpoint==0.3.2
- ordered-set==4.1.0
- pandas==2.2.1
- pandocfilters==1.5.0
- partd==1.4.1
- pathspec==0.12.1
- patsy==0.5.3
- pillow==10.0.0
- pip==24.0
- pluggy==1.4.0
- popv==0.2.2
- protobuf==4.24.0
- psutil==5.9.5
- ptxcompiler-cu11==0.8.1.post1
- pyarrow==14.0.2
- pyarrow-hotfix==0.6
- pyasn1==0.5.0
- pyasn1-modules==0.3.0
- pycparser==2.21
- pydantic==1.10.12
- pyjwt==2.8.0
- pylibcugraph-cu11==24.4.0
- pylibcugraph-cu12==24.4.0
- pylibraft-cu11==24.4.0
- pylibraft-cu12==24.4.0
- pynndescent==0.5.10
- pynvjitlink-cu12==0.1.14
- pynvml==11.4.1
- pyparsing==3.0.9
- pyro-api==0.1.2
- pyro-ppl==1.8.6
- pysocks==1.7.1
- python-editor==1.0.4
- python-multipart==0.0.6
- pytorch-lightning==2.0.6
- pytz==2023.3
- pyyaml==6.0.1
- pyzmq==25.1.1
- raft-dask-cu11==24.4.0
- raft-dask-cu12==24.4.0
- rapids-dask-dependency==24.4.1
- readchar==4.0.5
- referencing==0.30.2
- regex==2023.8.8
- requests==2.31.0
- requests-oauthlib==1.3.1
- rich==13.5.2
- rmm-cu11==24.4.0
- rmm-cu12==24.4.0
- rpds-py==0.9.2
- rsa==4.9
- s3fs==2023.12.2
- scanorama==1.7.3
- scanpy==1.9.3
- scikit-learn==0.24.2
- scipy==1.11.1
- scvi-tools==1.0.3
- seaborn==0.12.2
- secretstorage==3.3.3
- sentence-transformers==2.2.2
- sentencepiece==0.1.99
- session-info==1.0.0
- shellingham==1.5.4
- six==1.15.0
- sniffio==1.3.0
- snowballstemmer==2.2.0
- somacore==1.0.6
- sortedcontainers==2.4.0
- soupsieve==2.4.1
- sparse==0.14.0
- sphinx==4.3.2
- sphinxcontrib-applehelp==1.0.4
- sphinxcontrib-devhelp==1.0.2
- sphinxcontrib-htmlhelp==2.0.1
- sphinxcontrib-jsmath==1.0.1
- sphinxcontrib-qthelp==1.0.3
- sphinxcontrib-serializinghtml==1.1.5
- starlette==0.27.0
- starsessions==1.3.0
- statsmodels==0.14.0
- stdlib-list==0.9.0
- sympy==1.12
- tblib==3.0.0
- tensorboard==2.13.0
- tensorboard-data-server==0.7.1
- tensorflow==2.13.0
- tensorflow-estimator==2.13.0
- tensorflow-io-gcs-filesystem==0.33.0
- tensorstore==0.1.41
- termcolor==2.3.0
- texttable==1.6.7
- threadpoolctl==3.2.0
- tiledb==0.24.0
- tiledbsoma==1.6.1
- tinycss2==1.2.1
```

```
  - tokenizers==0.13.3
  - tomli==2.0.1
  - tomli-w==1.0.0
  - tomlkit==0.12.3
  - toolz==0.12.0
  - torch==2.0.1
  - torchmetrics==1.0.3
  - torchvision==0.15.2
  - tornado==6.3.3
  - tqdm==4.64.0
  - transformers==4.29.0
  - treelite==4.1.2
  - triton==2.0.0
  - trove-classifiers==2024.1.31
  - typing-extensions==4.2.0
  - tzdata==2024.1
  - ucx-py-cu11==0.37.0
  - ucx-py-cu12==0.37.0
  - umap-learn==0.5.3
  - urllib3==1.26.16
  - userpath==1.9.1
  - uvicorn==0.23.2
  - virtualenv==20.25.0
  - webencodings==0.5.1
  - websocket-client==1.6.1
  - websockets==11.0.3
  - werkzeug==2.3.7
  - widgetsnbextension==4.0.8
  - xarray==2023.7.0
  - yarl==1.9.2
  - zict==3.0.0
  - zipp==3.16.2
  - zstandard==0.22.0
prefix: /home/cane/miniconda3/envs/popv_bench
```

Data

Extended Human Lung Cell Atlas was downloaded from CELLxGENE: https://cellxgene.cziscience.com/collections/6f6d381a-7701-4781-935c-db10d30de 293. Lung Cell Atlas was downloaded from CELLxGENE https://cellxgene.cziscience.462 com/collections/Sd445965-6fla-4b68-ba3a-b8f765155d3a. Thymus data set was Extended Human Lung Cell Atlas was downloaded from CELLxGENE: https://cellxgene.cziscience.com/collections/6f6d381a-7701-4781-935c-db10d30de293. Lung Cell Atlas was downloaded from CELLxGENE https://cellxgene.cziscience.462 com/collections/5d445965-6f1a-4b68-ba3a-b8f765155d3a. Thymus data set was downloaded from Data was downloaded from https://cellxgene.cziscience.com/collections/de13e3e2-23b6-40ed-a413-e9e12d7d3910. Brain data set was downloaded from CELLxGENE https://cellxgene.cziscience/com/collections/283d65eb-dd53-496d-adb7-7570c7caa443. All data set are in the same form as the CELLxGENE census on 09/06/2023. After correction of metadata Tabula sapiens was deposited on Zenodo under DOI https://doi.org/10.5281/zenodo.7587774.

Human research participants

Reporting on sex and gender	N/A
Population characteristics	N/A
Recruitment	N/A
Ethics oversight	N/A

Field-specific reporting

Please select the one below that is the best fit for your research. If you are not sure, read the appropriate sections before making your selection.

[✓ Life sciences] Behavioural & social sciences Ecological, evolutionary & environmental sciences

For a reference copy of the document with all sections, see nature.com/documents/nr-reporting-summary-flat.pdf

Life sciences study design

All studies must disclose on these points even when the disclosure is negative.

Sample size	All query datasets contained samples from multiple human individuals (<10). This allows analysis of batch correction efficiency. All datasets contained less than 200k cells and more than 10k cells. This is a standard size for modern scRNA-seq datasets. We applied popV to larger datasets and discussed this in the response to the reviewers. This was critical to evaluate scalability.
Data exclusions	Cell-types with less than 10 cells were excluded as the K-nearest neighbor classifiers can't classify these (k=15 and so 8 cells are at least necessary). We excluded several donors from Tabula sapiens that had conflicting cell-type annotation. Within Tabula sapiens v2 we are currently refining these labels. This dataset is not publicly available yet.
Replication	All results are obtained from multiple human individuals. This demonstrates reproducibility. All findings are reproducible between these individual samples.
Randomization	No random subsetting was used. Query and reference dataset are coming from different sources.
Blinding	No blinding was performed as for every cell it was either unlabeled (query dataset) or labeled (reference dataset). The algorithm itself is blinded towards the existing query labels that were used to evaluate performance.

Reporting for specific materials, systems and methods

We require information from authors about some types of materials, experimental systems and methods used in many studies. Here, indicate whether each material, system or method listed is relevant to your study. If you are not sure if a list item applies to your research, read the appropriate section before selecting a response.

Materials & experimental systems

n/a	Involved in the study
✓	☐ Antibodies
✓	☐ Eukaryotic cell lines
✓	☐ Palaeontology and archaeology
✓	☐ Animals and other organisms
✓	☐ Clinical data
✓	☐ Dual use research of concern

Methods

n/a	Involved in the study
✓	☐ ChIP-seq
✓	☐ Flow cytometry
✓	☐ MRI-based neuroimaging