CNValidatron, automated validation of CNV calls using computer vision

Simone Montalbano[1], G. Bragi Walters[2], Gudbjorn F. Jonsson[2], Jesper R. Gådin[1], Thomas Werge[1,3], Hreinn Stefansson[2], Kári Stefánsson[2] & Andrés Ingason[1]

Institute of Biological Psychiatry, Mental Health Services, Copenhagen University Hospital, Roskilde, Denmark.
deCODE genetics/Amgen, Inc., Reykjavik, Iceland.
Department of Clinical Medicine, University of Copenhagen, Copenhagen, Denmark.

Abstract

Motivation

For more than a decade, running PennCNV on SNP array data has been the gold standard for detecting Copy Number Variants (CNVs, deletions and duplications). It is generally assumed that PennCNV has high sensitivity but poor specificity, leading to a large portion of CNV calls being false positives. Researchers often rely on manual inspection of the raw data to validate CNV calls. However, this approach is not feasible for more than a handful of loci in large collections.

Results

Here we present an R package implementing a convolutional neural network capable of automating CNV validation with an accuracy comparable to a trained human analyst. We also present an in-depth analysis into PennCNV false positive and false negative rates. Finally, we propose an algorithm to simplify the analysis of genome-wide CNV calls computing CNV regions.

Availability and implementation

The code is available on GitHub https://github.com/SinomeM/CNValidatron_fl.

Introduction

Copy Number Variants (CNVs) are a class of structural genetic variation, representing a segment of chromosomal DNA sequence that has been deleted, duplicated, or otherwise misplaced compared to the reference genomic sequence.[1] For the purposes of this manuscript, we define CNVs as all deletions and duplications ranging between small structural variants and large chromosomal rearrangements, roughly corresponding to ~50kb to ~10Mb.

For more than a decade, running PennCNV[2] on SNP array data has been the gold standard for detecting CNVs, especially in large collections of samples. While collections of whole-genome sequenced (WGS) samples are growing rapidly[3,4] and WGS (especially from long read technologies) is the ideal data source for detecting small and complex structural variants (SV), WGS is not necessarily better than SNPs arrays in detecting larger CNVs with good accuracy.

All CNV calling programs rely on some measure of relative DNA intensity (log-R-ratio (LRR) for SNP arrays and read depth for WGS data), and some also include a measure of allelic composition (such as B-allele frequency (BAF) for Illumina arrays)[5,6]. PennCNV uses a Hidden Markov Model to predict the state (copy number) of each marker, based on the state of the previous marker, integrating both LRR and BAF. Sets of consecutive markers with the same state are then stitched together to create the actual calls. CNV calling from SNP array data using PennCNV is believed to have a generally high sensitivity, however it tends to have poor specificity, with more than 50% of calls typically being false positives[7]. Depending on sample quality, array type and genomic location, the false positive rate can be much higher[7]. For this reason, researchers have used different approaches to control the false positive rate, including filtering whole samples as well as individual CNVs based on quality metrics, or using the intersection of calls across different CNV calling algorithms.[8] Moreover, especially when precision is extremely important (e.g. in very rare exposures such as recurrent CNVs)[9,10], visual validation, performed by a human analyst through inspection of LRR and BAF plots around the suspected CNV call, remains the best approach and is possibly second only to a wet-lab validation using a quantitative polymerase chain reaction (qPCR) experiment. Clearly, such an approach is overly time consuming, and its application is therefore limited to studies with a very specific focus, i.e., a handful of genomic loci.

In this study, we present a novel solution to this problem based on machine vision. We showcase an algorithm that is capable of automating the visual inspection of CNVs with an accuracy and precision comparable to (if not better than) that of a human analyst and distribute it as an R package.

This solution provides us with an unprecedented quality of a genome-wide CNV dataset, which in turn creates new research opportunities as well as new analytical challenges. One of the most impelling challenges being that CNVs are very heterogeneous, for example two CNVs that can be considered identical in practice while not having the exact same boundaries. In this case it is not trivial to decide when they can be treated as the same variant using a single simple rule. For this reason, we also developed a method to group CNVs into biologically plausible CNV regions (CNVRs) based on network analysis, and we demonstrate its function in a selected set of well characterised loci.

Methods

CNV calling and processing

In this study we use CNV calls from two different sample cohorts; a subsample of full trios (offspring, father, mother) from genotyped Icelandic samples at deCODE genetics (deCODE trios), and singleton samples from the UK Biobank (UKB). Samples were genotyped on different chips in different cohorts, Affymetrix UK Biobank Axiom Array for most UKB samples and multiple generations of Illumina arrays for the deCODE trios.

All CNV calls were generated using PennCNV[2]. The CNV calling in the Icelandic trios and UKB was performed at deCODE genetics. In brief, a PennCNV pipeline was used, main settings differing from default were minimum number of SNPs set to 5 and using a filtered SNP list as described below. After calling, all calls were processed using our CNV protocol.[11] In the deCODE data, samples had already been filtered for basic quality control (QC) measures (LRR_SD < 0.35, BAF_DRIFT < 0.01, GCWF between -0.02 and 0.02) while no QC filter was applied in the UKB. Then, individual samples were processed to stitch together consecutive calls if they had the same copy number and were close enough (gap not larger than 20% of the combined length). SNP markers were also filtered as previously described,[11] i.e. only biallelic autosomal SNPs mapping uniquely to the Haplotype Reference Consortium (HRC) hg19 reference map[12], with a minor allele frequency of at least 0.1%. Finally, samples with full or partial chromosomal abnormalities (loosely defined as any deletion or duplication larger than 25Mbp) were excluded. These were defined as chromosomes where CNV calls covered more than 10% of the total markers and were manually validated by a human analyst.

Training data

A schematic of the datasets used in the study is shown in figure 1. The model was trained on 13,078 CNV call examples from a collection of samples from the UKB and the Icelandic biobank at deCODE genetics obtained as follows: 15,000 samples were selected at random from the UKB cohort as well as 800 offspring from the deCODE trios. In the UKB data, a large portion of CNVs belonged to a small set of groups of CNVs (CNV regions, CNVRs). To avoid introducing bias towards specific genomic regions in the model, such regions were down-sampled as follows. CNVRs with a frequency above 20 were divided in three groups based on the frequency within the 15,000 samples: group A, CNVRs with a frequency up to 50, group B, CNVRs with a frequency between 50 and 100, and group C, CNVRs with a frequency above 100. Up to 20 CNVs were sampled from CNVRs in frequency group A, up to 35 from CNVRs in group B and up to 40 from CNVRs in group C.

All CNVs above 25 SNPs and 15 SNPs for UKB and deCODE examples respectively, were manually validated using in-house software[11] (https://github.com/SinomeM/shinyCNV) by at least one human analyst. Each CNV was classified as either true or false, leading to three possible categories: true deletion, true duplication, false call. Very ambiguous calls (e.g., a missed chromosomal aneuploidy) were allowed to be excluded from the training set, however, we tried to limit this as much as possible.

Prediction model design and training

The data representation was designed to be a simplified and condensed version of what the human analyst sees when performing the visual inspection of CNVs (i.e., tracks of LRR and BAF around a putative CNV in each sample). All the processing described here has two main objectives, first to fit all

necessary information in the smallest amount of data possible (here, the smallest image) to minimise computation, and second to make the CNVs representation as similar as possible across different genomic loci, different CNVs sizes, different markers numbers and distributions (to minimise any influence of indirect factors on the evaluation of each call). We loosely tested different resolutions before setting it to 96 by 96 pixels. As shown in supplementary figure 1, each image is composed of three rows, containing two views of the LRR values around the CNVs and one view of the BAF. Horizontally the image is composed of three sectors: markers on the left, within and to the right of the CNV call. Depending on the size of the CNV call (below 100kbp, between 100kbp and 1Mbp, above 1Mbp) the two bottom rows will cover 19, 15 or 11 CNVs lengths in total, with equal spacing on each two side, while the top row is always zoomed out by a further factor of three. This is done so that the number of markers in the image is more consistent across CNV length ranges.

The pixel image is constructed as follows. For each of the three rows, all points are converted to the coordinate system, i.e., from genomic position in x and LRR/BAF in y to 0-96 in x and [a, b] in y (where a and b values depend on the specific row). In this smaller space, multiple markers will fall onto the same pixel and the counts for each individual pixel, n, are collected. Then, n is first moved to the log scale and then to [0, 1]. This is done in windows to minimise the effect of differential SNPs density across the image, that could otherwise drown the signal of normal regions, due to small segments of very high SNPs coverage (such as exons). Window size is 4, 8 or 32 pixels, depending on the CNVs length class, as described above. LRR values are processed before being converted to the pixel values system to reduce the effect of outliers. This process is done independently for the three sectors of the image: left side of the CNV, inside the CNV or right side of the CNV. In brief, the farther a point is from the mean value of its sector the more it is 'pulled' towards it, as in the following R pseudo-code (for values above the mean): `new_lrr = lrr-abs(lrr-mean_lrr)*shrink_lrr_factor`, with `shrink_lrr_factor` set to 0.2 by default. In practice, this has very little effect on most points, but should mitigate the effect that outliers might have on the final image. This is done to simulate what the human eye does automatically during visual inspection, i.e., give more "weight" to the centre of the distribution and ignore outliers, if rare. This process does not alter the overall shape of the distribution, especially any waviness pattern that might be present. After this step, any remaining outlier, defined as a point outside three standard deviations of the sector distribution, is removed.

We increased the number of examples using data augmentation in the form of image flipping on the x axis with a probability of 0.5 per example. Other types of data augmentation, such as adding random noise to each pixel, were considered but eventually discarded. Training was performed on the whole dataset in a 75-25% configuration for training and validation set.

The model was built using torch[13] with the R packages torch[14], torchvision[15] and luz[16]. It is composed of 5 convolutional layers (`nn_conv2d`) followed by a 10-layer linear classifier (`nn_linear`). Each layer has a 5% dropout rate (`nn_dropout2d(p=0.05)`). Training was performed using `luz::fit()` with the following settings: max learning rate of 0.2, max epochs of 50, and early stopping with a patience value of 4.

Test data

Model accuracy was tested in a different set of deCODE samples, while precision was tested in both UKB and deCODE data. The UKB set was designed to also test the calibration of the model probability output as follows. 2800 predicted true CNVs were randomly selected, sampling 400 CNVs from each of the following prediction probability intervals [0.75, 0.85), [0.85, 0.90), [0.90, 0.92), [0.92, 0.94),

[0.94, 0.96), [0.96, 0.98), and [0.98, 1]. In the deCODE test set, all CNVs from 1200 randomly sampled individuals were selected. For the UKB calibration set all selected CNVs were manually validated by a human analyst and tagged as either true, false or undetermined/unknown (i.e., calls that could not be ruled out as being true while lacking strong indication of being so). For the purpose of computing accuracy and precision, unknown calls were considered as false. For the deCODE test set we partially relied on the genealogy for validation, and we considered as true any CNVs predicted true by the program and present in at least one of the carrier's parents with at least 50% overlap without the need of manual visual inspection. All discordant CNVs (i.e., those predicted as true by the program but where no concordant call was present in a parent or vice-versa) were manually validated. Accuracy is defined as (TP + TN) / (TP + FP + TN + FN), where TP, TN, FP, and FN are true positives, true negatives, false positives and false negatives respectively, with the truth being the human analyst evaluation for UKB, and both model-genealogy agreement and human evaluation in the deCODE sets.

CNVR detection algorithm

The CNVR detection algorithm is composed of three separate steps. First, CNVs are divided into networks, defined as the smallest set of mutually overlapping CNVs. By definition, the largest possible network is thus a chromosomal arm (in our pipeline CNVs cannot span across the centromere), and the smallest network is a single CNV that does not overlap with any neighbour. The second step consists in creation of the CNVRs. For each network, an NxN matrix is constructed, where N is the number of CNVs in the network and each entry is the *intersection over the union* (IOU) between a pair of CNVs. Two identical CNVs will have an IOU of 1, two overlapping CNVs will have an IOU between 0 and 1, depending on the similarity, and two non-overlapping CNVs will have an IOU closer to -1 the more distant they are. This matrix is then translated into a weighted `igraph`[17] network object where all pairs with an IOU above a certain minimum value will be connected, with the IOU values being the strength of the connection. Default minimum IOU value is set to 0.75. Higher minimum values tend to create smaller, tighter networks, while lower minimum values tend to create larger and potentially more shallow networks. We do not recommend setting the minimum IOU value below 0.5. This network is then processed with the community detection algorithm Leiden[18] (`igraph::cluster_leiden()`, default resolution value is set to 1), and the resulting networks are converted back into CNVRs with the boundaries being the median values of the start and end of the constitutive CNVs respectively. CNVs with no CNVR (singletons) are given their own group. The third and final step is an optional forced merge of CNVRs. For each chromosome, all CNVRs are processed in order of start position. For any given CNVR x, all regions with a reciprocal overlap above a certain threshold with x (default value is 0.75) are merged, the new start and end are computed again as the median from all the CNVs in the new group. This process continues until all CNVRs have been either processed or merged with a previous one. This is done for a fixed number of iterations, 5 by default. Each CNVR can be modified only once per iteration, thus the higher the number of iterations, the more CNVRs will be merged. The idea is to combine very similar CNVRs, especially spurious results from the community detection algorithm, without moving the boundaries of the region too far away from the corresponding CNVs.

When computing CNVRs on genome wide CNVs, we defined three groups of settings to test, very strict, loose, and in between, defined as follows. Very strict (`min_iou=0.9`, `max_force_merge_rounds=1`, `force_merge_min_overlap=0.9`) maximise intra-CNVR similarity, loose (`min_iou=0.5`, `max_force_merge_rounds=5`, `force_merge_min_overlap=0.75`) minimise the number of regions created, while the settings

for the in-between group (`min_iou=0.75`, `max_force_merge_rounds=3`, `force_merge_min_overlap=0.75`) should be a balance between the two extremes.

Software availability

The program is available as an R package at https://github.com/SinomeM/CNValidatron_fl/. The visual validation interface is available at https://github.com/SinomeM/shinyCNV. Most of the data processing was done using the R package QCtreeCNV available at https://github.com/SinomeM/QCtreeCNV and previously described.[11]

Results

PennCNV false positive calls

In this study we used CNV calls from two different cohorts, genotyped on different arrays from two manufacturers to investigate and mitigate the burden of false positive CNV calls in large datasets. As described in the methods, these consist of UK Biobank (UKB)[19] and deCODE trios (deCODE genetics / Amgen, Reykjavík, Iceland).

We assessed the PennCNV false positive rate in three different CNV sets from two cohorts. These consist of 1) UKB training set, 2) deCODE training set, 3) deCODE test set. 15,000 samples were randomly selected in the UKB and 2,000 offspring from full trios in the deCODE sets (800 train and 1,200 test respectively), for a total of 17,000 unique samples. All CNVs from these samples were processed and filtered for 25 minimum number of SNPs in the UKB and 15 in deCODE data (table 1). All CNVs were manually validated for the deCODE portion of the training set, while common CNVRs were down sampled in the UKB training set (see methods). This was done to reduce the model bias towards specific common loci and was deemed necessary only in the UKB portion of the training set. Finally, CNVs in the deCODE test set were validated only in the case of discordance with the genealogy, as described in the methods. In total almost 22,168 CNVs were validated (12,153 UKB train, 3,760 deCODE train, 6,255 deCODE test). CNVs were evaluated by at least one human analyst and tagged as either true or false. Results are shown in table 1. In summary, ~30% to ~60% of the CNV calls produced by PennCNV are false positives depending on the specific cohort and the minimum number of SNPs.

Prediction model training and testing

The model training was performed on a combined set of human-validated CNV calls from UKB and deCODE, for a total of 13,078 unique examples. Training stopped after 12 epochs with an accuracy of 0,96.

Accuracy and precision are 95% and 92% in the deCODE test set respectively (table 2), while precision is 95% in the UKB calibrations set. We also tested the calibration of the probability outputted by the model in the UKB. We randomly selected a total of 2,800 predicted true CNVs from intervals of probabilities. As shown in supplementary figure 4, the output probability is not properly calibrated, but can be used as a lower limit, meaning that selecting CNVs predicted true with at least 0.75 probability will lead to a dataset where at least 75% of CNVs would be evaluated as true by a human analyst. In practice this percentage will likely be much higher (95% in this within-sample set).

PennCNV false negatives estimation and *de novo* CNV rate

In contrast with false positives, false negatives are essentially impossible to quantify without other means of detecting CNVs. We attempted to get an estimate of the false negative rate using the deCODE trios from our analysis of the calls where the model prediction was not supported by parent-offspring segregation consisting of 637 CNVs, excluding the MHC region. Of these, 99 were found to be true positives in the offspring and false negatives in the parent, meaning that the CNV was present (based on visual inspection) but had been missed by PennCNV in the parent. The majority of these (96 of 99) are duplications and the median length is ~90kbp, with 11 being larger than 200kbp. This would set the minimum false negative CNV call rate at ~1.5% (99 out of 6522 CNVs) and ~4.0% (96 out of 2413) for duplications only.

238 Moreover, 42 CNVs (30 deletions, and 12 duplications) were found to be true in the offspring but not
239 present in the parents with a median length of ~161kbp, setting the minimum *de novo* CNV mutation
240 rate at ~3.5% (42 events in 1,200 samples).

241 False positives calls and QC measures

242 We investigated the link between false positive PennCNV calls and quality measures, LRR standard
243 deviation (LLR_SD), BAF drift, and GC waviness factor (GCWF). As shown in supplementary figure 3,
244 false CNVs (especially deletions) tend to be from samples with somewhat higher LRR SD and BAF drift,
245 but not GCWF. As expected, undetermined CNVs lie in between true and false ones. Moreover, true
246 CNVs (especially duplications) tend to be larger than false and undetermined ones (supplementary
247 figure 2). True duplications also tend to have a higher number of markers than any other categories.
248 This is consistent with our experience, as duplications tend to need more points to be reliably
249 considered true evaluated by the analyst compared to deletions.
250 We also compared our program accuracy against standard QC filtering from previous literature in the
251 UKB train set, as Kendall et al.[20]. Excluding samples with a |WF| > 0.03, more than 30 CNVs or LRRSD
252 > 0.35. In the UKB train set, this excluded 144 (2%) true, 1,163 (54%) false and 122 (6%) undetermined
253 CNVs, removing 24% of the CNVs evaluated as not true (false + undetermined) by a human analyst
254 while sacrificing only 2% of the true positives.

255 Differences in distribution between true and false positive calls

256 To understand the effect that insufficient filtering of false positive CNVs can have on a genome wide
257 analysis, we plotted the genomic distribution of the UKB train set, marking human validated true and
258 not-true CNVs (supplementary figure 5). Similarly, to the overall CNVs distribution, differences in true
259 and not-true distribution vary widely both within and across chromosomes. In the UKB train set (panel
260 a), the distribution of true and false CNV calls is very similar for chromosomes 16 and 18, and 5 and 9
261 for deletions and duplications respectively. In contrast, they are very different in chromosomes 9 and
262 20 for deletions, and 11 and 20 for duplications. In the deCODE test set (panel b), we did not identify
263 a chromosome where the distribution of true and false CNVs followed each other as closely as in the
264 UKB train set. In contrast the two distributions are very different in chromosomes 8, 13 and 17 for
265 both deletions and duplications.

266 CNV regions algorithm

267 CNV regions (CNVRs) do not have a unanimous definition in the literature. In this study we defined
268 them as the results of grouping CNVs based on similarity, maximising the within-group similarity while
269 minimising the number of different groups. We tested multiple algorithms based on incrementally
270 growing CNVRs, that is add a new CNV to a group if it is similar enough to its members (e.g., in terms
271 of reciprocal overlap or intersection over the union, IOU), or create a new group if no match is found.
272 We found that such approaches tend to be slow and more importantly, give potentially very different
273 results based on the order in which CNVs are evaluated. For this reason, we developed an algorithm
274 to compute CNVRs in a whole locus at the same time, based on the community detection method
275 Leiden, from network analysis.
276 We applied our CNVR detection algorithm to all CNVs predicted as true deletions or duplication with
277 a probability above 75% in the UKB (~525,000). We tested three sets of settings (very strict, in-
278 between, and loose) and evaluated the results at four well characterised genomic loci: the *NRXN1*

gene locus on the 2p arm, and the recurrent CNV loci 15q11.2-13.3, 16p13.1-p12.3, and the 22q11.2 broader region (figure 2 and supplementary figure 6). The most stringent setting (green in supplementary figure 6) creates too many groups, given the high internal similarity requirements, and it is not shown in figure 2. Between the two, neither is preferable in all possible situations. As shown in figure 2, some CNVRs seem to be very specific, meaning that all CNVs within them are extremely similar, and are detected with the same number of carriers in both methods. Two examples demonstrating this are the 127 carriers CNVR in 15q13.2 marked with '*' (figure 2 panel a) and the 30 carriers one in 16p13.11 marked with '~' (figure 2 panel b). However, in most cases the two settings will produce slightly different results. The looser setting tends to create fewer CNVRs that are less specific, meaning CNVs in them can have less closely aligned boundaries. In figure 2 we annotated two such cases in known CNV loci. In 15q13.3, with loose settings there is a CNVR including 2,815 carriers overlapping the known CHRNA7 copy-number-variable region marked with '+' that is separated into three different regions with 2,281, 484 and 168 carriers respectively when applying the strict/intermediate setting. Conversely, in 16p13.11 there are 756 carriers of a CNVR marked with '#' covering the known recurrent locus that the medium setting correctly separates in two groups, the most frequent including an additional group of markers on the right (Mb 15.0-15.2), whereas the loose setting groups the two CNV subtypes together in one CNVR. Depending on the application, researchers can balance the number and carrier frequency of CNVRs with their specificity in terms of internal similarity.

Discussion

In this study we provide the first in-depth analysis of the extent of false positive CNV calls from PennCNV across two cohorts genotyped on different SNP arrays from two different manufacturers. We demonstrate that false positives constitute a large fraction of calls across cohorts, while depending somewhat on factors such as array type (Illumina or Affymetrix), sample quality, and the number of probes in each CNV call. Moreover, we show that standard QC filtering practice only remove a minor portion of the false positives in UKB data. Finally, we show that false and true CNV calls do not follow the same genomic distribution. Studies that do not use a strong method to exclude false positive calls (such as visual inspection) will therefore likely produce highly biased results.

Of note, PennCNV produced fewer CNV calls on Affymetrix arrays compared to Illumina, both before and after filtering. Possible reasons for this include differences in markers density between the two manufacturers, lower signal to noise ratio in Affymetrix data, and lower sensitivity for Affymetrix data from the PennCNV model. While using only one algorithm (PennCNV) to call CNVs, we do not think this is a strong limitation as, 1) PennCNV is by far the most widely used software to call CNV in SNP array data, 2) PennCNV has been shown to be the most reliable software in a previous comparison[23] and 3) most programs (with the notable exception of ipattern[21]) uses the same raw data (LRR and BAF) and some implementation of the same HMM architecture as PennCNV.[5,22,23]

We also provide, to our knowledge, the first estimate of false negative rate in CNV calls. Given the high number of false positives, it is normally assumed that PennCNV is over-sensitive, and that false negatives are not an issue. Indeed, we find them to be most likely less than the false positives, with an estimate from the deCODE trios' data setting the minimum to 1.6%, and mostly confined to smaller duplications. Considering most CNVs are rare or ultra-rare exposures the false negative rate might still be a source of bias in some contexts, and particularly difficult to mitigate *in-silico*, especially without access to full pedigrees, as is the case in most modern biobanks.

Visual inspection remains the best solution to the false positives in PennCNV calls, and arguably the most reliable. However, it is extremely time consuming, and remains virtually unfeasible for genome wide applications in large collections. Here, we provide software to automate CNVs validation using computer vision, with an accuracy comparable to a trained human analyst, across multiple datasets.

In our view, the major limitation of this work is the absence of an "absolute truth" to compare with. Both the training examples and the test sets are based on visual inspection by a human analyst, and this leads to possible biases. Even though a wet-lab validation using qPCR experiment would be the *ideal* gold standard, visual inspection remains the *practical* gold standard in the field, and the best option available on such a scale and given the sensitivity of the data. Moreover, having access to a full pedigree, allowing to compare the same region across closely related samples, reduces uncertainty in human evaluation.

We also propose an algorithm to compute CNV regions. Also in this case, there is no clear "gold standard" nor unanimous definition of what such regions should look like or how they should behave genome-wide. For our purposes we define CNVRs as "the smallest possible number of sets that can group CNVs that are as similar as possible together". Clearly, low number of groups and internal similarity cannot be maximised at the same time. We show how different settings of our algorithm can produce different "flavours" of CNV regions. Strict setting will create few but more heterogenous groups, which might be ideal to limit the number of variants to test in an associations study, with loose setting will produce several but more homogenous regions, which might be ideal when the focus is on precise genomic structures such as a specific gene. We showcase the results of our algorithm at

different well-characterised loci, and we believe some degree of manual scrutiny is required when computing CNVRs, however it's impossible to manually check the entire genome for coherence.

In conclusion, despite some important but mostly out-of-our-control limitations, we believe the software described is capable of removing the vast majority of false positives from a CNV calls set genome wide, and not introduce strong false negative issues. This will open the field to new and interesting research opportunities.

Bibliography

1. Freeman, J. L. *et al.* Copy number variation: new insights in genome diversity. *Genome Res.* **16**, 949–961 (2006).
2. Wang, K. *et al.* PennCNV: An integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. *Genome Res.* **17**, 1665–1674 (2007).
3. Taliun, D. *et al.* Sequencing of 53,831 diverse genomes from the NHLBI TOPMed Program. *Nature* **590**, 290–299 (2021).
4. Halldorsson, B. V. *et al.* The sequences of 150,119 genomes in the UK Biobank. *Nature* **607**, 732–740 (2022).
5. Seiser, E. L. & Innocenti, F. Hidden Markov Model-Based CNV Detection Algorithms for Illumina Genotyping Microarrays. *Cancer Inform.* **13s7**, CIN.S16345 (2014).
6. Panda, A. *et al.* Genome-wide analysis and visualization of copy number with CNVpytor in igv.js. *Bioinforma. Oxf. Engl.* **40**, btae453 (2024).
7. Winchester, L., Yau, C. & Ragoussis, J. Comparing CNV detection methods for SNP arrays. *Brief. Funct. Genomic. Proteomic.* **8**, 353–366 (2009).
8. Marshall, C. R. *et al.* Contribution of copy number variants to schizophrenia from a genome-wide study of 41,321 subjects. *Nat. Genet.* **49**, 27–35 (2017).
9. Vaez, M. *et al.* Population-based Risk of Psychiatric Disorders Associated with Recurrent CNVs. 2023.09.04.23294975 Preprint at https://doi.org/10.1101/2023.09.04.23294975 (2023).
10. Crawford, K. *et al.* Medical consequences of pathogenic CNVs in adults: analysis of the UK Biobank. *J. Med. Genet.* **56**, 131–138 (2019).
11. Montalbano, S. *et al.* Accurate and Effective Detection of Recurrent Copy Number Variants in Large SNP Genotype Datasets. *Curr. Protoc.* **2**, e621 (2022).

12. McCarthy, S. *et al.* A reference panel of 64,976 haplotypes for genotype imputation. *Nat. Genet.* **48**, 1279–1283 (2016).
13. Ansel, J. *et al.* PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. in *29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)* (ACM, 2024). doi:10.1145/3620665.3640366.
14. Falbel, D. & Luraschi, J. torch: Tensors and Neural Networks with 'GPU' Acceleration. (2024).
15. Falbel, D. torchvision: Models, Datasets and Transformations for Images. (2024).
16. Falbel, D. luz: Higher Level 'API' for 'torch'. (2024).
17. Csárdi, G. *et al.* igraph for R: R interface of the igraph library for graph theory and network analysis. Zenodo https://doi.org/10.5281/zenodo.10681749 (2024).
18. Traag, V. A., Waltman, L. & van Eck, N. J. From Louvain to Leiden: guaranteeing well-connected communities. *Sci. Rep.* **9**, 5233 (2019).
19. Sudlow, C. *et al.* UK biobank: an open access resource for identifying the causes of a wide range of complex diseases of middle and old age. *PLoS Med.* **12**, e1001779 (2015).
20. Kendall, K. M. *et al.* Cognitive performance and functional outcomes of carriers of pathogenic copy number variants: analysis of the UK Biobank. *Br. J. Psychiatry* **214**, 297–304 (2019).
21. Pinto, D. *et al.* Functional impact of global rare copy number variation in autism spectrum disorders. *Nature* **466**, 368–372 (2010).
22. Colella, S. *et al.* QuantiSNP: an Objective Bayes Hidden-Markov Model to detect and accurately map copy number variation using SNP genotyping data. *Nucleic Acids Res.* **35**, 2013–2025 (2007).
23. Seiser, E. L. & Innocenti, F. Hidden Markov Model-Based CNV Detection Algorithms for Illumina Genotyping Microarrays. *Cancer Inform* **13s7**, CIN.S16345 (2014).

Figures legends

Figure 1: Schematic representation of the training and test sets across the two cohorts.

Figure 2: IGV plot showing CNVRs computed with different settings. From the top, the light grey track shows the cytobands, the dark grey show known recurrent CNV loci, and the second dark grey shows the SNP markers coverage. The red track shows CNVRs computed with the looser setting, while the blue one shows the medium settings. The strict setting track is shown only in supplementary figure VII in green. The number below each CNVR is the number of CNV belonging to it. a) 15q13.3 b) 16p13.1-12.3. CNVRs with a frequency below 10 are not shown.

Figures

Figure 1.

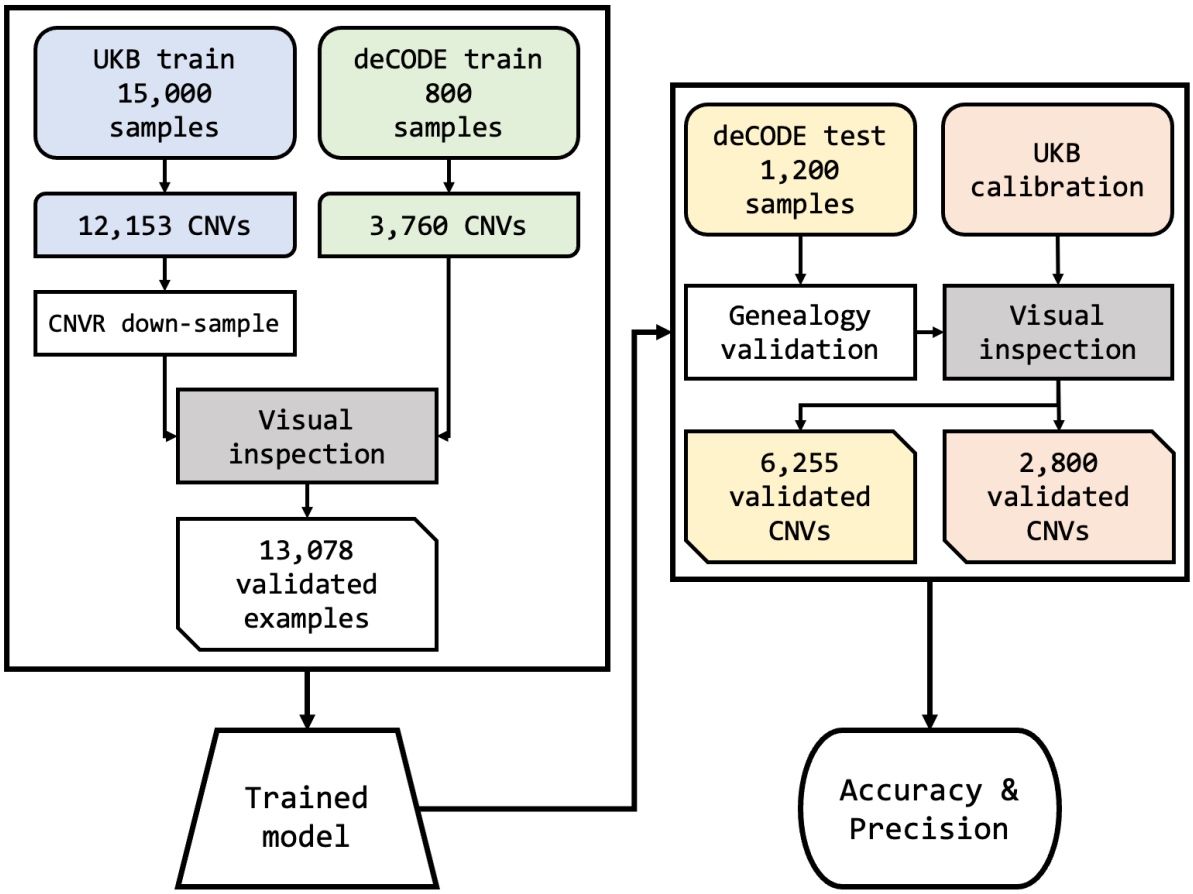
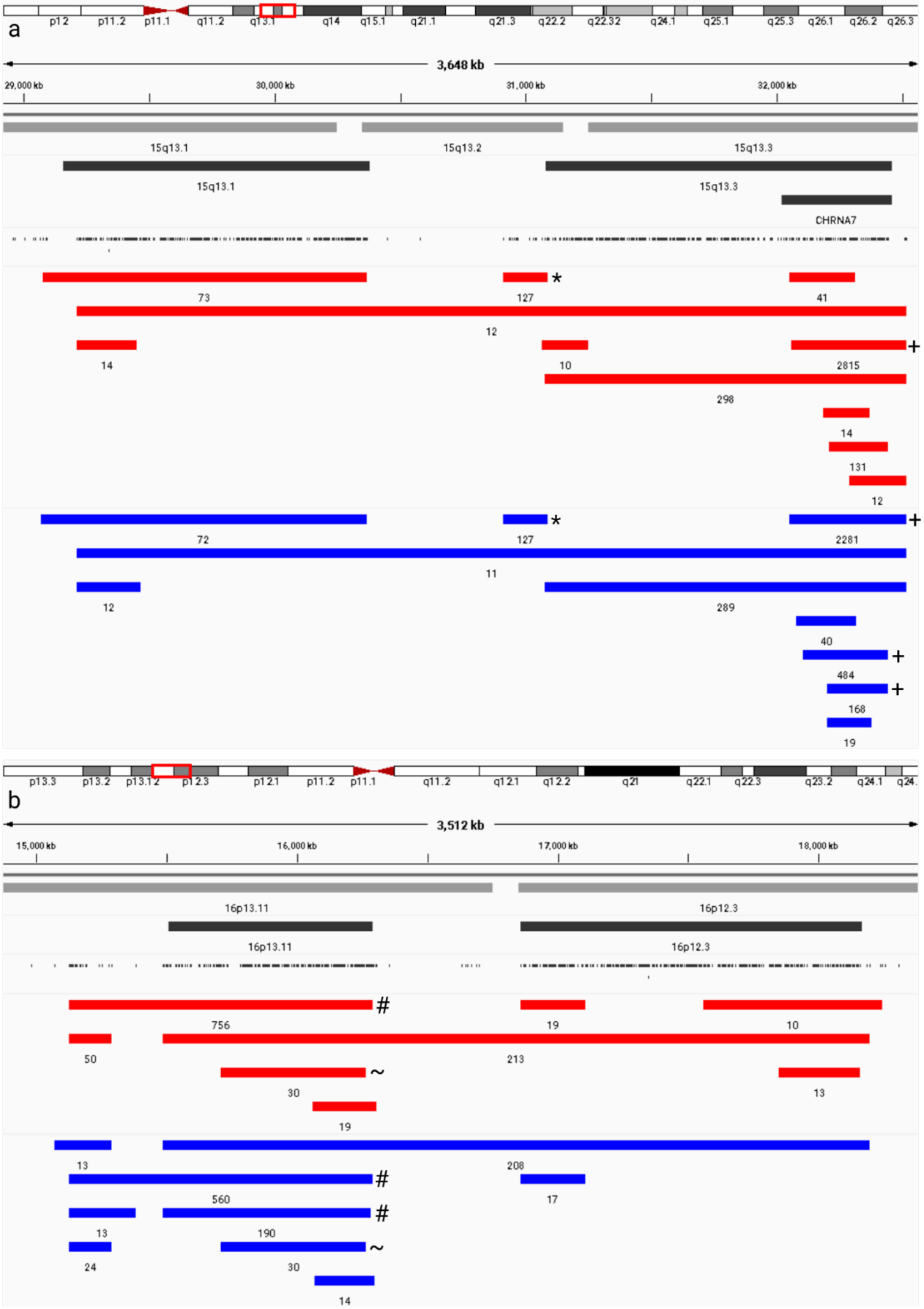


Figure 2.


a
p12
p11.2
p11.1
q11.2
q13.1
q14
q15.1
q21.1
q21.3
q22.2
q22.32
q24.1
q25.1
q25.3
q26.1
q26.2
q26.3
3,648 kb
29,000 kb
30,000 kb
31,000 kb
32,000 kb
15q13.1
15q13.2
15q13.3
15q13.1
15q13.3
CHRNA7
73
127
41
12
14
10
2815
298
14
131
12
72
127
2281
11
12
289
40
484
168
19
b
p13.3
p13.2
p13.12
p12.3
p12.1
p11.2
p11.1
q11.2
q12.1
q12.2
q21
q22.1
q22.3
q23.2
q24.1
3,512 kb
15,000 kb
16,000 kb
17,000 kb
18,000 kb
16p13.11
16p12.3
16p13.11
16p12.3
756
19
10
50
213
30
13
19
13
208
560
17
13
190
24
30
14

Tables legends

Table 1: Results from human visual validation in the different CNV sets. Training and test sets consist of all CNVs from a randomly selected set of samples, with some filters applied.

Table 2: Accuracy and precision in the different CNV sets.

Tables

Table 1.

Dataset	Chip	Min length	Min SNPs	Tot samples	Tot CNVs	CNV/ Sample	True (%)	False (%)	Unknown (%)
deCODE train[1]	Multiple	NA	15	800	3760	4.7	1547 (41%)	2213 (59%)	0 (0%)
deCODE test[1]	Multiple	NA	15	1200	6522	5.4	2790 (43%)	3732 (57%)	0 (0%)
UKB train[2]	Affymetrix	NA	25	15,000	12153	0.8	6829 (56%)	3296 (27%)	2028 (17%)
UBK calibration	Affymetrix	NA	15	NA	2800	NA	2665 (95%)	5 (0%)	130 (5%)

[1]: for deCODE data, multiple full trios were inspected together, this essentially eliminated the need for the undetermined/unknown tag.

[2]: common CNVRs were down sampled in this set.

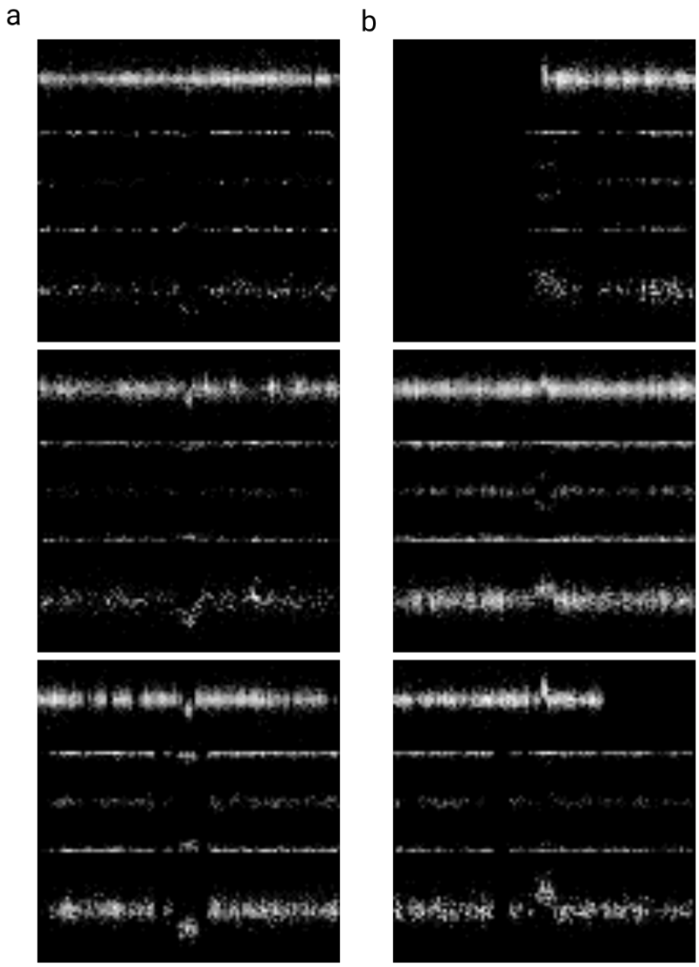
426 **Table 2**.
427

Dataset	**TP**	**TN**	**FP**	**FN**	**Accuracy (TP+TN)/All**	**Precision TP/(TP+FP)**
deCODE test	2578	3639	93	212	95%	92%
UBK calibration	2666	NA	134	NA	NA	95%

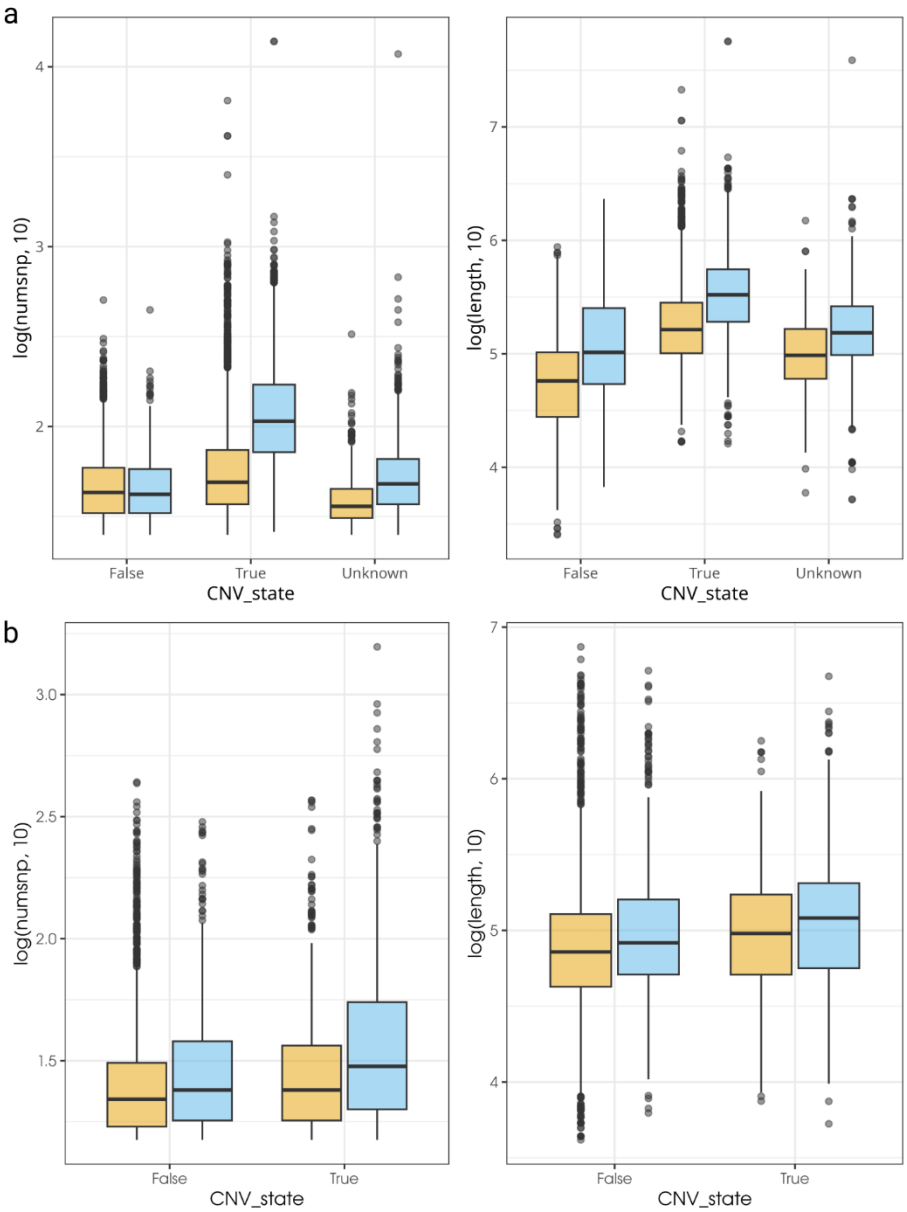
428

Supplementary material

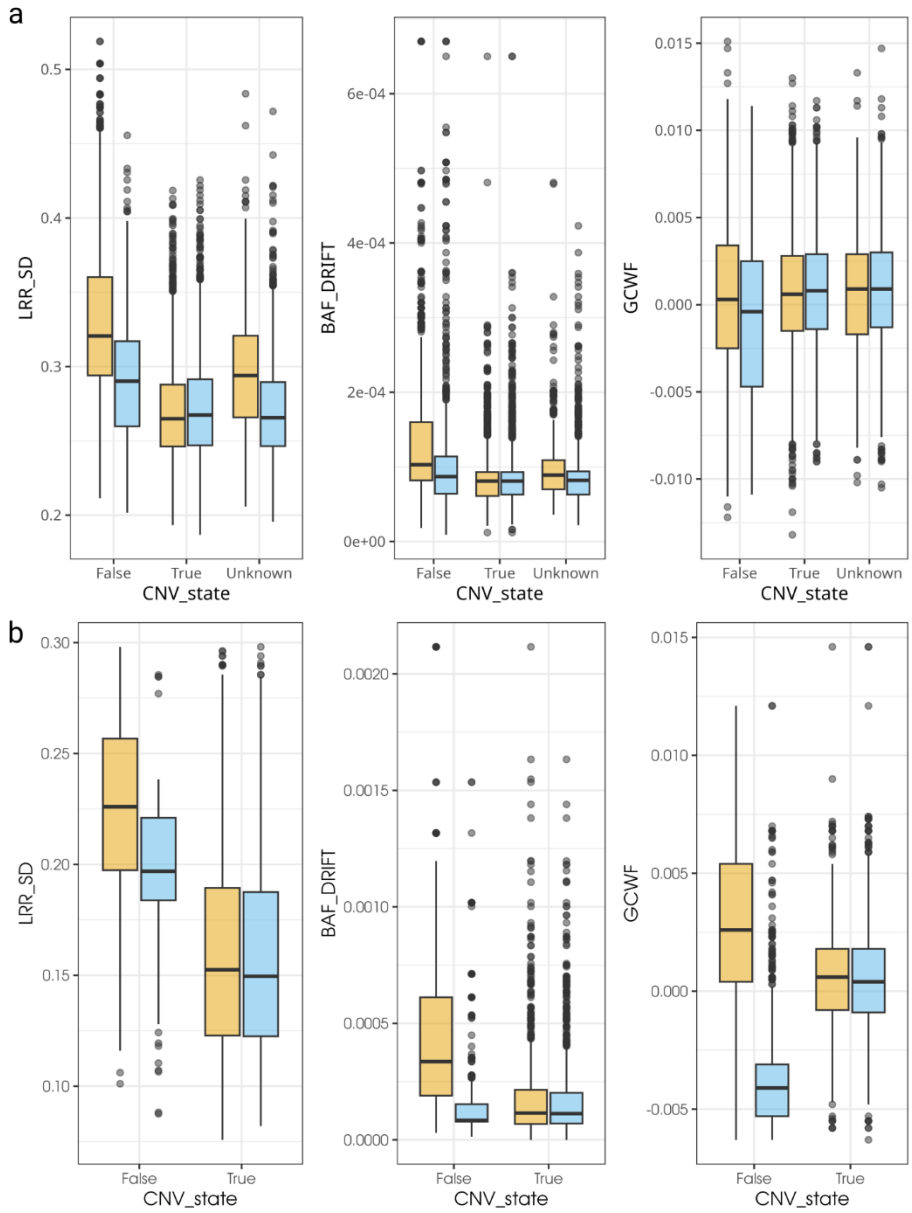
Supplementary figure 1. Examples of the 96x96 pixels image for the CNN. Three examples of true deletions are shown in a, and three examples of true duplication in b. In each image it can be recognised the three data rows described in the methods, form the top, a zoomed-out view of the LRR values around the CNV call, the BAF in the middle, and the LRR values on the bottom. The deletions showcase the characteristic “dip” in LRR value and a loss of heterozygosity in the BAF (especially clear in the bottom example), while the duplications showcase the increase in LRR values and the split of central BAF band (especially clear in the first and second from the top).



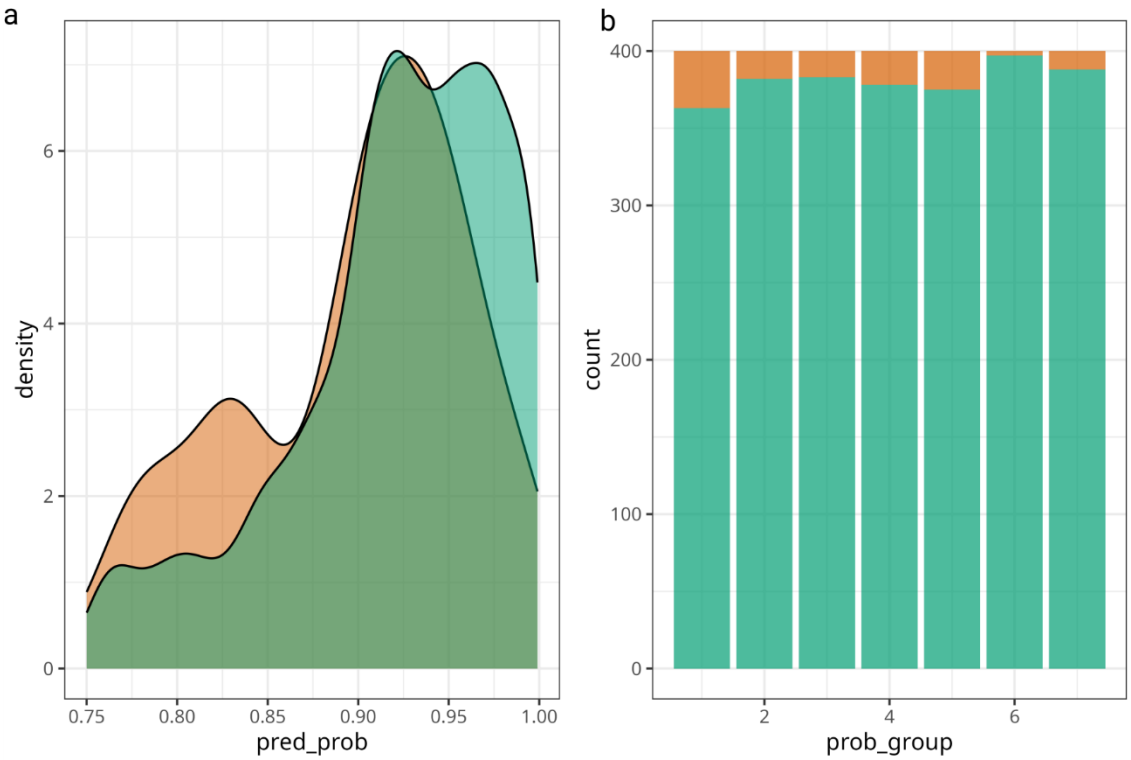
Supplementary figure 2. Boxplots showing the distributions of number of markers and length of CNVs across the different categories: deletion/duplication, true/false/unknown. Log 10 number of SNP markers (left) and length (right) in the true, false and undetermined/unknown CNVs. Deletions in yellow, duplications in blue. a) UKB train, b) deCODE test. Duplications tends to be larger and have more markers than deletions, especially in UKB data. True CNVs also tend to be larger and have more markers than false/unknow ones, again, especially in UKB.



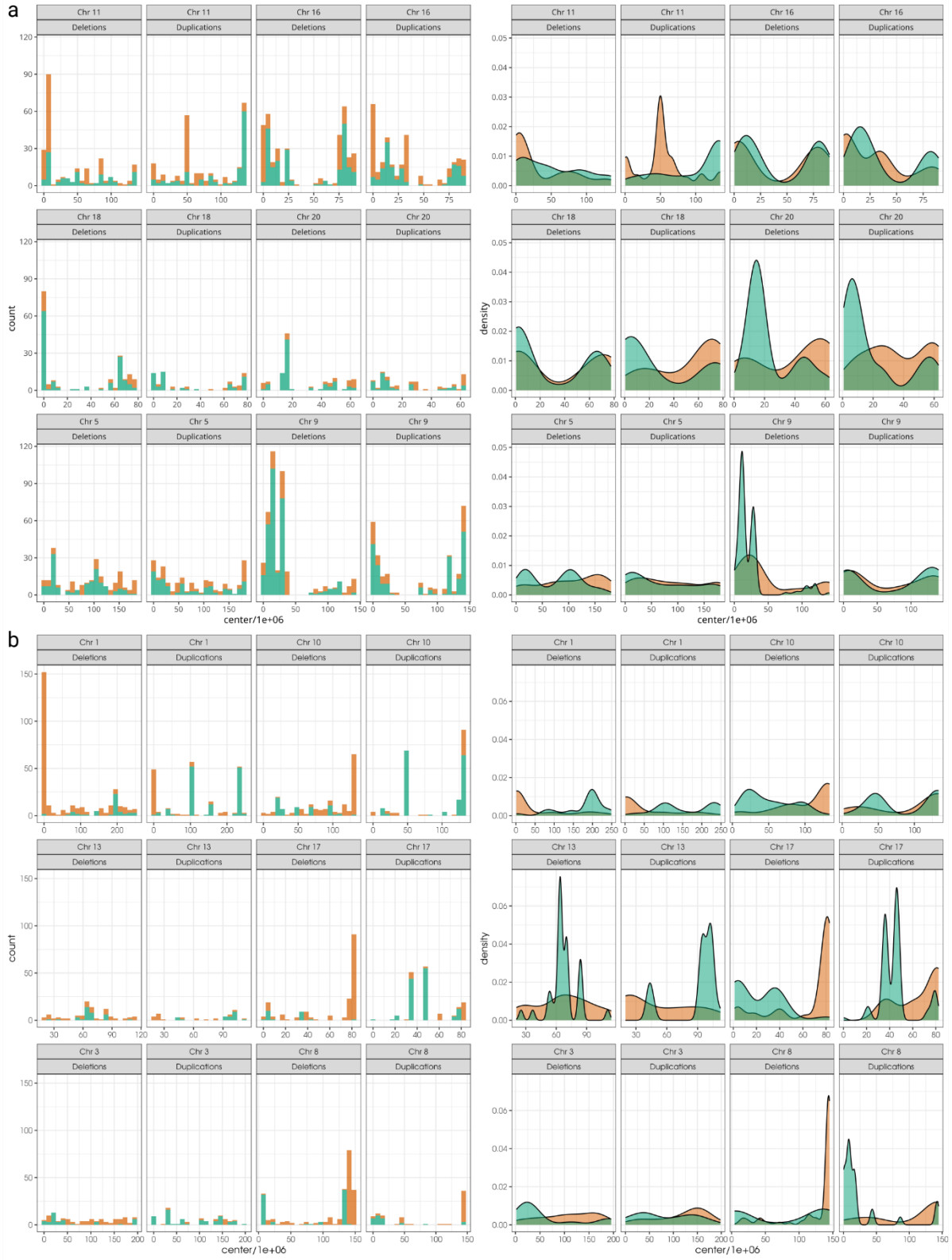
Supplementary figure 3. Boxplots of the three main QC measures across the different categories: deletion/duplication, true/false/unknown. LRR SD (left), BAF drift (middle), and GCWF (right) distributions in the true, false and undetermined/unknown CNVs. Deletions in yellow, duplications in blue. Each CNV inherits the QC measure of the sample. a) UKB train, b) deCODE test. False CNVs tend to belong to samples with high LRR SD in both UKB and deCODE data, the same is true for false deletions and BAF drift. GCWF seems to be different only for false deletions in deCODE data.



Supplementary figure 4. Results of Visual validation in the UKB calibration set. All CNVs have been predicted as true by the model and have been validated by a human analyst. 400 CNVs were sampled for each probability group, as follows. 1: [0.75, 0.85), 2: [0.85, 0.90), 3: [0.90, 0.92), 4: [0.92, 0.94), 5: [0.94, 0.96), 6: [0.96, 0.98), and 7: [0.98, 1]. a) overall distribution of the prediction probability in the true (green) and false (orange) CNVs. b) bar plot of the visual validation status in each prediction probability group. Lower groups seem to have a higher proportion of false CNVs, but there is no clear overall pattern.



Supplementary figure 5. Genomic distribution of the True and False CNV calls in the a) UKB train set, and b) deCODE test. True CNVs in green, false in orange. Deletion and duplications are plotted separately for a selection of chromosomes, both stacked histograms, on the left, and density plots, on the right, are provided. In both plots type, the x axes show the genomic coordinate of the centre position of CNVs, while on the y axes show the count for each bin and the density for the staked histogram and density plot respectively. In the density plots, the density for true and false CNVs is computed separately, so the histogram shows the overall CNVs distribution better, while the in density plots it is easier to see the individual distribution of the two groups. The plots show how in some chromosomes true and false CNVs seem to reflect the same distribution, while in some other chromosomes they follow completely different patterns.

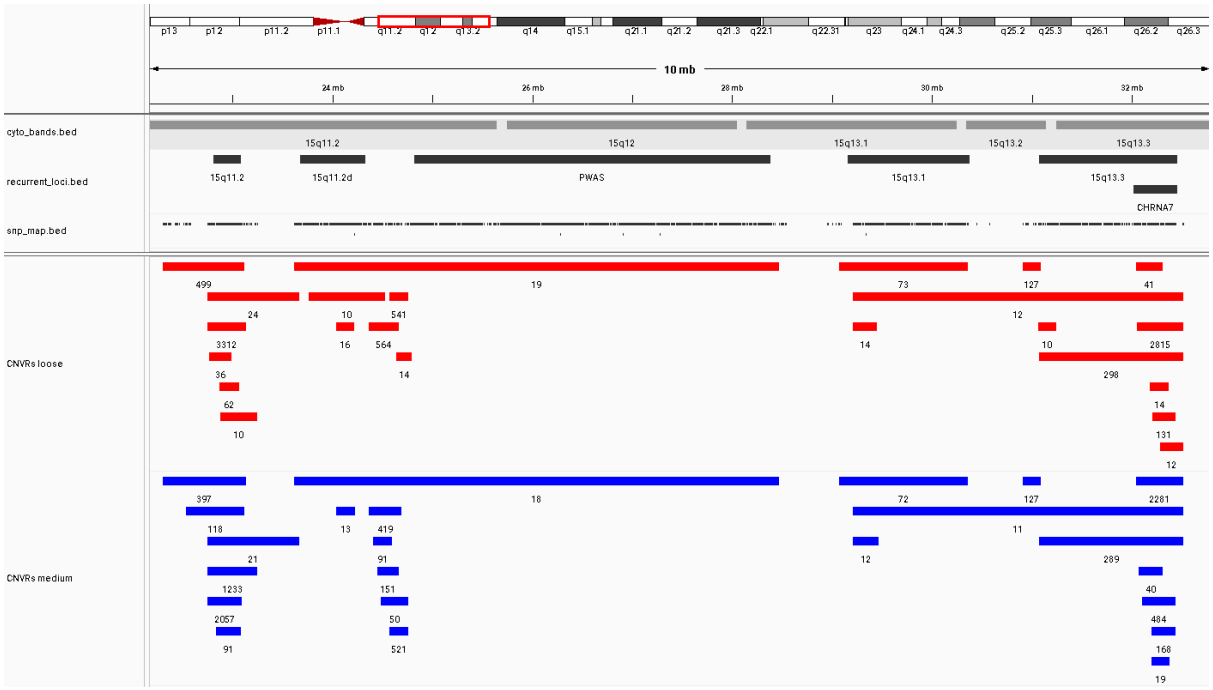

a
Chr 11
Deletions
Duplications
Chr 16
Chr 18
Chr 20
Chr 5
Chr 9
count
density
center/1e+06
b
Chr 1
Chr 10
Chr 13
Chr 17
Chr 3
Chr 8

475

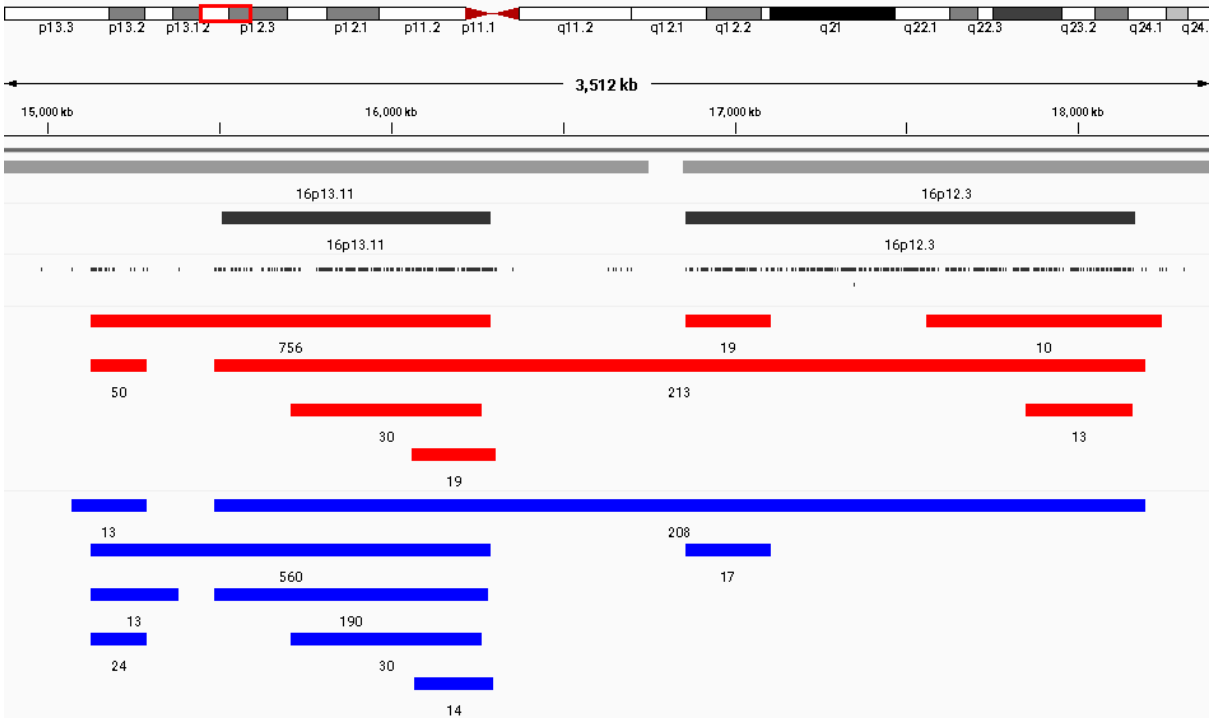
Supplementary figure 6. Showcase of CNVRs algorithm results. This is an extended version of figure 2. CNVRs computed with different settings. Red: loose, blue: medium, green: strict. Four different genomic loci are shown: a) *NRXN1* gene locus (chr2:49,645,643-51,759,674), b) 15q11.2-13.3 (chr15:22,600,000-32,600,000), c) 16p13.1-12.3 (chr16:15,000,000-18,300,000), d) 22q11.1-11.2 (chr22:17,200,000-23,800,000). For the sake of space, the green track is shown only in panels a and d. This is due to the high number of CNVRs produced by the strict setting. Only regions with a minimum frequency of 10 are plotted, so this effect is not present in panel a and only slightly present in panel d, where the number of total CNVs is lower than panels b and c and thus most small CNVRs are not plotted.

a)

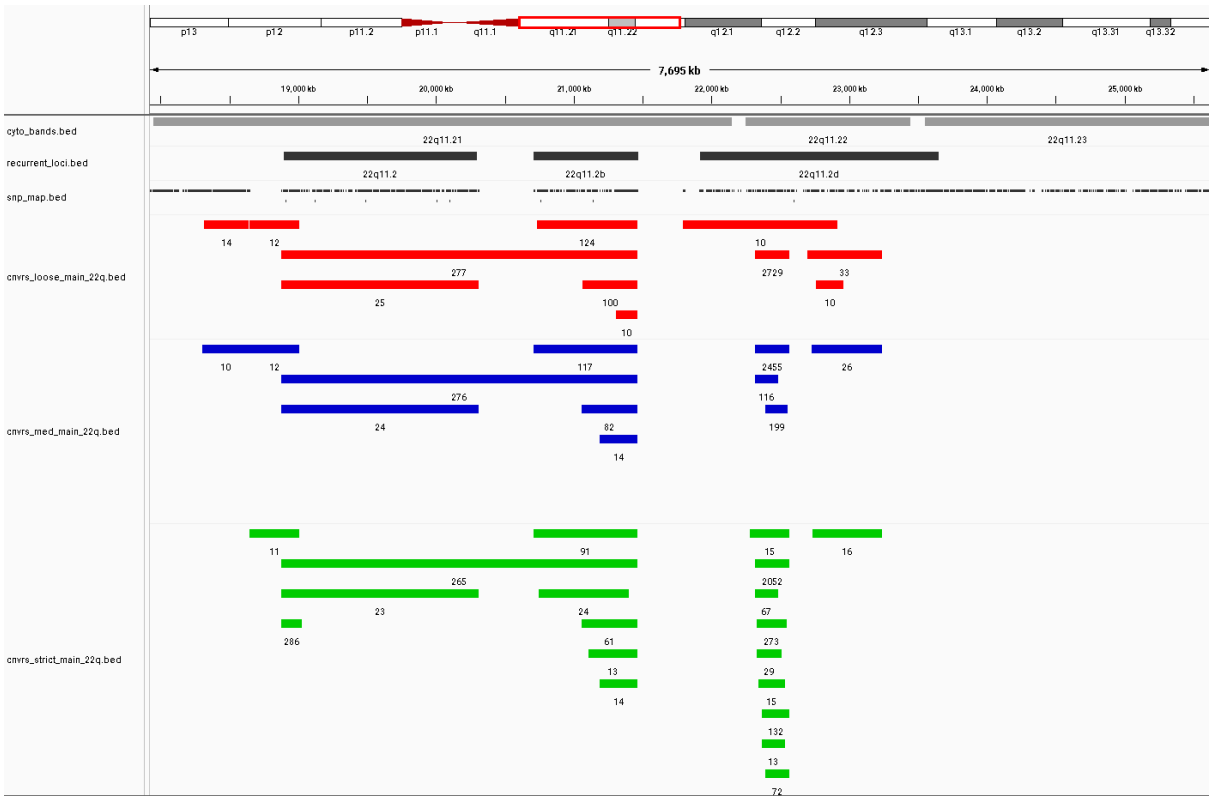
b)


10 mb
cyto_bands.bed
recurrent_loci.bed
snp_map.bed
15q11.2
15q11.2d
15q12
PWAS
15q13.1
15q13.2
15q13.3
CHRNA7
CNVRs loose
CNVRs medium

c)


3,512 kb
15,000 kb
16,000 kb
17,000 kb
18,000 kb
16p13.11
16p12.3

d)


p13
p12
p11.2
p11.1
q11.1
q11.21
q11.22
q12.1
q12.2
q12.3
q13.1
q13.2
q13.31
q13.32
7,695 kb
19,000 kb
20,000 kb
21,000 kb
22,000 kb
23,000 kb
24,000 kb
25,000 kb
cyto_bands.bed
22q11.21
22q11.22
22q11.23
recurrent_loci.bed
22q11.2
22q11.2b
22q11.2d
snp_map.bed
cnvrs_loose_main_22q.bed
14
12
124
10
277
2729
33
25
100
10
10
cnvrs_med_main_22q.bed
10
12
117
2455
26
276
116
24
82
199
14
cnvrs_strict_main_22q.bed
11
91
15
16
265
2052
23
24
67
286
61
273
13
29
14
15
132
13
72