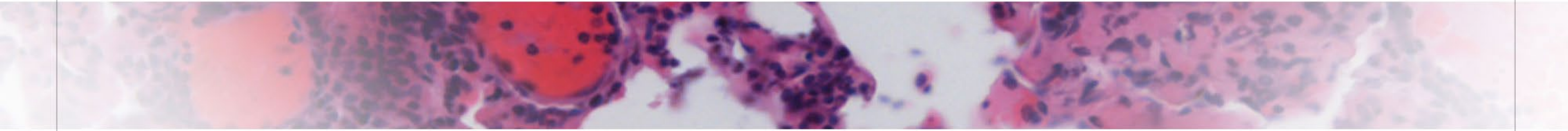
American Society *of* Hematology
Helping hematologists conquer blood diseases worldwide



A highly sensitive and specific gene fusion algorithm based on multiple fusion callers and an ensemble machine learning approach

K Brad Thomas, PhD[1*], Yanglong Mou, PhD[1*], Lauryn Keeler[2*], Christophe Magnan, PhD[3*], Vincent Funari, PhD[4*], Lawrence Weiss, MD[4*], Shari Brown, MD[5*] and Sally Agersborg, MD, PhD[4*]
[1]NeoGenomics Laboratories, Inc., Houston, TX; [2]NeoGenomics, Carlsbad, CA; [3]NeoGenomics Laboratories, Inc., Carlsbad, CA; [4]NeoGenomics Laboratories, Inc., Aliso Viejo, CA; [5]Neogenomics, Inc., Aliso Viejo, CA

Disclosure:

- All authors of this work are/were employees of NeoGenomics Incorporated.
- Funding for this work was entirely by NeoGenomics Incorporated.

Abstract

- **Background:** Gene Fusion events are common occurrences in malignancies, and are frequently drivers of malignancy. FISH and qPCR are two methods often used for identifying highly prevalent gene fusions/translocations. However, these are single target assays, requiring a lot of effort and sample if multiple assays are needed for multiple targets like sarcoma. High-throughput parallel (NextGen) DNA and RNA sequencing are also in current use to detect and characterize gene fusions. RNA sequencing (RNAseq) has the advantage that multiple markers can be targeted at one time and RNA fusions are readily identified from their product transcripts. While many fusion calling algorithms exist for use on RNAseq data, sensitive fusion callers, needed for samples of low tumor content, often present high false positive rates. Further, there currently is no single variable or element in NGS data that can be used to filter out false positive calls by extant callers. Individual sensitive fusion callers may be considered weak predictors of gene fusions. Combining their results into a single fusion call involves evaluating many elements, which can be a time consuming and difficult manual task. In order to achieve higher accuracy in fusion calls than can be achieved using individual fusion callers, we have combined the results of multiple fusion callers by use of an ensemble learning approach based on random forest models. Our method selects the best group of callers from among several callers, and provides an algorithmic means of combining their results, presenting a metric that can be immediately interpreted as the probability that a called fusion is a true fusion call.
- **Methods:** Random forest models were generated with the randomForest package in R, and then tuned using the R caret package. Training data sets consisted of fusion calls deemed true by review and by orthogonal methods including PCR/Sanger sequencing and the commercial Archer™ fusion calling system. We present the results of training on calls made by five fusion callers Arriba, STAR-Fusion, FusionCatcher, deFuse, and kallisto/pizzly. Logistic training variables (seen vs not seen by the fusion caller) were used for the five callers. Variables also included metrics for the magnitude and balance of coverage on either side of candidate fusion breakpoints reported by Arriba and STAR-Fusion ("coverage balance") and a single metric consisting of the number of sequencing reads that cross the candidate breakpoint. The model was validated by 10-fold cross-validation on 598 fusion calls by the five callers.
- **Results:**
- The resulting model is superior to the simple strategy of requiring agreement by n of five callers, particularly with regard to specificity (Table 1). Also, "importance of variables," reported by randomForest, gauges the relative contribution of variables in the model. Here it shows that one caller, kallisto\pizzly, does not contribute to the model (Table 2).
- **Conclusion:** Random Forest modeling provides a viable means of combining gene fusion call data from multiple callers into a single fusion calling tool with improved performance over simple combinations of fusion calls. An additional benefit is seen in that building and evaluating such models can guide the selection of fusion callers, thereby eliminating non-contributory calling methods and ensuring optimal utilization of computational resources.

Background

- The outstanding need our work is intended to address is for sensitive and specific fusion calling from RNA sequencing (RNAseq) data.
- Limiting false-positive and false-negative calls is critical for high-throughput fusion calling. Manual review is time-intensive, and must be assisted by appropriate algorithms in order to ensure reasonable turnaround time.
- Available fusion caller suites and algorithms were found to be sensitive, but also prone to high false-positive rates.
- We chose an ensemble learning approach as a defensible, objective and systematic means of weighting and combining metrics provided by multiple off-the-shelf fusion callers. The result is a single metric – the probability of membership in one of two classes: true and reportable, not true or not reportable.

Methods (1)

- RNAseq data obtained using an in-house developed QIAseq-based panel.
- We initially used five distinct fusion-calling pipelines:
  - Arriba
  - STAR-Fusion
  - FusionCatcher
  - deFuse
  - kallisto/pizzly
- Along with logistic variables – yes/no the fusion caller called the candidate fusion – we also gathered underlying metrics from the callers. These were
  - chimeric read counts associated with the fusion junction
  - discordant read pairs that straddle the fusion junction
  - depth of coverage on each side of the fusion junction

Methods (2)

- The metrics and logistic variables were used as features for modeling
- Software for modeling consisted of the R application *randomForest* which was run within the *caret* optimization environment.
- Training employed the default 500 trees (ntrees) with 10-fold cross-validation.
- Data comprised 598 candidate fusion calls – each made by at least one of the five fusion callers.

Results (1)

- In a proof-of-concept study, we compared a random forest classifier with the simpler approach of only requiring that n of the five callers make the call.

	Random Forest Model	Require 5 callers	Require at least 4 callers	Require at least 3 callers
Accuracy (Acc)	0.9849	0.9833	0.9666	0.8127
95% CI	(0.9716, 0.9931)	(0.9695, 0.992)	(0.9488, 0.9795)	(0.7791, 0.8432)
No Information Rate (NIR)	0.9532	0.9532	0.9532	0.9532
P-Value [Acc > NIR]	0.00002111	0.00006407	0.06804	1
Kappa	0.8223	0.7743	0.6973	0.2683
Mcnemar's Test P-Value	0.505	0.004427	0.00365	<2e-16
Sensitivity	0.78571	0.64286	0.89286	0.96429
Specificity	0.99474	1	0.97018	0.80526
Pos Pred Value	0.88	1	0.59524	0.19565
Neg Pred Value	0.98953	0.98276	0.9946	0.99783

Variable	Importance
Star Fusion coverage balance	100
Arriba coverage balance	86.965
Star Fusion	73.923
Reads across breakpoint	56.661
DeFuse	44.279
Arriba	20.179
Fusion Catcher	8.992
Kalisto/pizzly	0

Results (2)

- The successful initial proof-of-concept has led to further work. We have improved on modeling with resulting improved sensitivity. Our current model, applied to hematologic malignancies, requires the three independent fusion callers
  - Arriba
  - STAR-Fusion
  - FusionCatcher

  We have also implemented a fusion junction counter that is independent of the three fusion callers, and included its results as a feature in the model.

Accuracy	0.9653
95% CI	(0.9351, 0.984)
No Information Rate	0.5058
P-Value [Acc > NIR]	<2e-16
Kappa	0.9305
Mcnemar's Test P-Value	1
Sensitivity	0.9618
Specificity	0.9688
Pos Pred Value	0.9692
Neg Pred Value	0.9612

Discussion

We have arrived at a random forest model that can accurately classify candidate fusions as true or false fusions.

Increased sensitivity has brought with it new challenges. Our current focus is on discrimination between driver fusions and “benign” fusions.

References

- Arriba software https://github.com/suhrig/arriba
- STAR-Fusion software https://github.com/STAR-Fusion/STAR-Fusion
- FusionCatcher software https://sourceforge.net/projects/fusioncatcher
- deFuse software https://github.com/amcpherson/defuse
- kallisto software https://pachterlab.github.io/kallisto
- pizzly software https://github.com/pmelsted/pizzly
- R application caret https://cran.r-project.org/web/packages/caret/caret.pdf
- R application randomForest https://cran.r-project.org/web/packages/randomForest/randomForest.pdf