Computational methods of genetic engineering detection: a review and recommendation for future work

J. Vivian Belenky, September 2022

Bottom Line Up Front

- Existing genetic engineering detection (GED) work, including unpublished FELIX content, has been focused on plants and prokaryotes, and largely on genetic isolates, not large microbial communities. Gaps remain in metagenomic GED and in viral GED.
- Barriers to advancing GED capabilities include:
    - Absence of a high quality test dataset. The creation of this dataset should be a top priority.
    - Quality of reference data. Most publicly available databases of 'natural' sequences, even reference databases, are hopelessly contaminated with engineered sequences. Usage of any of these will require extensive expert curation prior to use.
    - Quality of metagenomic assembly. Most forms of agnostic GED are limited by the quality of the sequencing assembly. Current assembly algorithms are not ideal for either metagenomic or viral assembly.
    - Characterization of surveillance sites. Without a reasonably thorough characterization of a surveillance site, it will be difficult to establish the probability of any given novel sequence being either engineered or naturally evolved.
- Trade-offs in various GED techniques are inevitable. Truly agnostic techniques are reliant on the fragility of reference databases and assembly algorithms, and are not particularly fast. Fast, semi-agnostic techniques have limited sensitivity. Fast, sensitive techniques rely on DNA signatures and are vulnerable to red-teaming.
- Opportunities exist for the advancement of various GED sub-modules. However, there is a risk that these approaches will not readily transfer to metagenome-derived data.
- At least proximally, GED is not a machine learning problem, but a bioinformatics problem. Opportunities to leverage machine learning in this context must be integrated with existing bioinformatic tools, and informed by biological knowledge. Any machine learning component of a GED system must also maintain interpretability as a top priority.

Table of Contents

Introduction

The purpose of this document is to establish a state of the art of computational genetic engineering detection, with particular focus on methods relevant to agnostic metagenomic sequencing.

Agnostic genetic engineering detection capabilities are critically important for early detection of high-concern biothreats, as well as for the detection of accidental release of modified organisms from laboratories working with high-concern biothreats.

A substantial literature exists on genetic engineering detection–however, the vast majority of this work is focused on detecting genetic engineering in plants[1–3]. The overwhelming majority of this work focuses on experimental techniques, which typically involve the development of workflows for the sensitive detection of known markers of engineering, and has generally been performed on purified single species (isolates) rather than complex microbial communities.

The majority of recent computational work is unpublished material from the IARPA FELIX project, where four major performers focused on bioinformatic and machine learning methods of detection. I summarize these methods, as well as other computational work, and their limitations, and make recommendations for future work.

IARPA FELIX

FELIX (**F**inding **E**ngineering **L**inked **I**ndicators) was an IARPA project announced August 31st, 2017, aimed at developing both wet and dry techniques for the detection of genetic engineering, which concluded in the spring of 2022 after 40 months. The vast majority of the information in this section was learned over interviews taking place largely over the summer of 2022. Cleaned up transcripts and notes from these calls, as well as other communications, may be found here.

Project goals

FELIX's original broad agency announcement (BAA) was left intentionally vague, with performers left to determine almost everything about the project parameters for themselves–what 'counts' as engineering, what approach to use, what milestones to aim for, and so on. Part of FELIX's goal was for performers to feel out how to think about the problem of genetic engineering detection, as well as to establish a state of the art. Intended deliverables included platforms, approaches to sample analysis, and computational tools and models.

Testing and evaluation

There were four total rounds of testing and evaluation, over two phases. Phase 1 was the 'proof of concept' phase, with evaluations taking place at month 8 and month 16, after which there was a round of down-selection. Performers selected to continue entered Phase 2, with additional rounds of evaluations at month 24 and month 40. After each round of evaluation, performers were given 30 days to analyze their performance and "explain themselves."

In the first two rounds, testing and evaluation procedures were unique to each performer's approach and focus, with less than a quarter of test sets being shared between performers. The third round had up to half the samples shared among performers, and the last was the same for every performer–it was also the most challenging.

Evaluation challenges included physical samples as well as sequencing data, which included both short and long reads. Most samples were either single isolates or mixed-organism samples. These samples varied along dilution level down to one engineered read in one million, and sequencing coverage ranging from 30x to less than 1x.

Common organisms included in the clonal samples included *Escheria coli, Bacillus subtilis* and *Saccharomyces cerevisiae.* Other organisms included *Citrobacter freundii,* a common hospital-acquired infectious agent in the immunocompromised, and *Pseudomonas aeruginosa* and *Pseudomonas putida,* included to serve as mutual confounders and for their relevance as biofilm-creating agents.

The final, challenge-mode round of testing and evaluation (T&E) included a gut microbiome metagenomic sample, an organism with over 800 components of engineering, and an influenza virus with a single stop codon deleted. The final metagenomic sample was spiked with engineering in silico, allegedly due to time constraints, suggesting that most of the other samples were created in a lab and then sequenced.

Results

By the end of the program, performers were reliably able to detect most kinds of engineering in single or mixed-organism cases, and were able to detect foreign gene content ("obvious" engineering) in at least some large, complex metagenomic samples. I was able to discover that Ginkgo performed at about 70% accuracy on the final, consistent dataset, while Raytheon BBN achieved 67% sensitivity and ~95% specificity—but without any information about the nature of how these metrics were calculated and what exactly went into the T&E dataset, it's difficult to make any hard conclusions.

Performers struggled with highly complex samples and especially with dilute metagenomic samples. Subtler forms of engineering, such as the deletion of a single stop codon in an influenza virus genome, were similarly difficult to detect, and in some cases were not achieved by any performer. Determination of the method of engineering was in some cases also impossible.

Successful computational techniques generally involved some form of ensembling, such as consensus alignment, gradient boosting, and random forest. Unsurprisingly, the combination of multiple approaches for the 'flagging' of a sequence as engineered outperform any specific detection technique. However, ensembling necessarily trades off against computational efficiency.

Several performers initially attempted deep learning approaches, but found their results to be either comparable or worse to standard bioinformatic techniques. Few if any performers included neural networks in their final pipelines.

As of late 2022, all FELIX computational pipelines have been transitioned to at least three government agencies: DTRA, DEVCOM, and NBACC. As of December 2024, at least DEVCOM was still in the process of downloading the tools and getting them to run (see CBD S&T 2022 poster here). Reportedly, DEVCOM is mostly only interested in long reads deployable in the field, and is primarily constrained by the sheer size of the pipelines and the computational resources required to run them.

Limitations

Program design

A limitation of the program was a perceived lack of clarity around its goals. There was a sense that FELIX did not have a particular final stakeholder they were targeting–which would normally be another government agency with a specific, concrete goal. System scalability didn't become a project goal at all until halfway through the program.

Because the initial program announcement was left intentionally vague and open-ended, testing and evaluation was perceived to be a moving target; performers often did not know what exactly they were trying to do, and were having to do it on very short (~8 month) time scales, often only with access to inadequate data. This may partly be a result of the fact that original FELIX PM, Amanda Dion-Shultz, left the project for unclear reasons partway through the program.

Another consequence of the program design was the difficulty in comparing methods to each other, at least for an external observer. Early rounds of T&E were unique to performers, and later rounds featured particularly non-central, adversarial examples of engineering. It is therefore hard to say with confidence which approach would perform best in the agnostic pathogen surveillance context where efficiency is key. It is unclear if or when the testing and evaluation data will be made available by IARPA.

Finally, FELIX was explicitly primarily interested in detecting engineering in isolates. This is ultimately reflected in the final outputs; relatively little time was spent on solving the metagenomics GED problem, even by performers who were interested in it.

Performers

For performers, a major limitation was data. Several performers attempted to use the NCBI database as a source of natural reference genomes, and found it to be hopelessly contaminated with improperly labeled engineered sequences. All performers ended up having to do extensive data curation, often duplicating one another's efforts; some performers ultimately gave up on establishing what was 'natural' using databases. Even RefSeq, intended to be a curated database of only reference and representative genomes, was found to be polluted, as many

reference genomes derive from lab strains which are engineered to make them more convenient for analysis.

(It should be noted that while IARPA did set up a shared server for performers to share data and support one another's efforts, almost nobody ended up actually using this shared server, and collaboration was minimal. It is unclear why this was the case.)

The contamination of data sources contributed to the ultimate underperformance of machine learning techniques, which are necessarily example-based. The undetected and difficult to locate presence of engineering in natural databases made the establishment of "natural" examples impossible. Performers attempted multiple techniques for mitigating this, such as by filtering out any sequences deposited by synthetic biologists and by other information contained in sequence metadata. This was variably successful and often insufficient, due in part to the untrustworthy nature of the metadata itself.

Secondly, there were serious challenges with establishing a sufficiently robust example-base of engineered sequences, particularly in a metagenomic or purely computational context. Several performers cited issues in comprehensively simulating the systematic error modes of different sequencing machines. One performer, Noblis Inc., profiled the error rate on their own in-house machines, and used the *E. coli* taxon on GenBank as their engineered rather than natural examples. However, they did not end up simulating any metagenomic data, which comes with its own challenges.

Thirdly, generated examples often failed to generalize; for instance, a model trained on a slew of modifications made to *S. cerevisiae* did not work at all on other organisms. Finally, many machine learning model forms are difficult to interpret; given known issues with generalizability, it is imperative that any model that spits out a yes-or-no judgment about the engineering status of an unknown sequence be interpretable by a human.

Summary of Performers' Approaches

Raytheon BBN - GUARDIAN

Project lead: Nicholas Roehner
Program manager: Susan Katz
Principal investigators: Joel S. Bader, Mona Singh, Eric M. Young, Dennis Eastburn, Adam Abate

The Raytheon BBN effort–*GUARDIAN: Guard for Uncovering Accidental Release, Detecting Intentional Alterations, and Nefariousness*–included multiple subgroups, inside and outside of Raytheon. Some of these were inside the company, but several were academic collaborators, including the Bader group at Johns Hopkins University and the Singh group at Princeton, with Eric Young at Worcester Polytechnic Institute brought in to serve as the expert on yeast engineering.

The top-level approach was an ensemble of the consortium's constituent parts. All teams would process the T&E data and return the sequences of interest. These sequences would then be aligned with each other; a sequence of interest was considered 'real' if at least two performers agreed.

Overall, by combining all performer's approaches with a multiple sequence alignment consensus algorithm, the Raytheon BBN consortium was able to achieve 90% sensitivity and 95% specificity *for foreign inserts,* with the primary limitation being an inability to detect insertions smaller than 30 base pairs. On the overall final dataset, specificity was similar, but sensitivity fell to about 67%.

Nic reportedly has plans to eventually publish his team's curated natural database.

Internal Raytheon BBN

Team: Aaron Adler, Jacob Beal, Daniel Wyschogrod, Stavros Tsakalidis, Fusun Yaman

Internal approaches varied significantly. One approach was extremely basic; it simply did a BLAST search and returned a sequence as 'of interest' if any keywords related to genetic engineering were found in the metadata of the returned sequences. This approach didn't work very well, presumably in part due to the extremely low quality of the annotations in the NCBI database.

On the other end of the spectrum, internal Raytheon ML teams tried adapting an existing neural network architecture originally used for identifying computer viruses. It is unknown what specific architectures were deployed, but it was generally assumed by Raytheon BBN that genetic engineering detection was an ideal use-case for deep learning. At least some of these models were trained on examples of heavily engineered yeast. However, after the first round of testing and evaluation, these models performed surprisingly poorly.

Bader Group - JHUardian

Principal investigator: Joel S. Bader (Johns Hopkins University)
Team: Peter Ge, Jitong Cai, Roy Siegelmann

Joel S. Bader was initially brought in only to consult as a biologist experienced with yeast engineering and did not participate in the first round of T&E. However, after the poor performance of the deep learning models, there was a concern that the Raytheon group would be "voted off the island" if they did not perform better on the second round.

Joel's approach was read-first; it flagged individual reads before assembly and follow-up analysis of contigs. A number of versions of his group's pipeline, JHUardian, were created. For single organism or few-organism mixtures, the reads would be aligned to reference genomes, and only the reads which did *not* align perfectly to a known reference would be retained for assembly and annotation. For the metagenomics case, reads would instead be aligned to the Univec database of known engineering markers, and only those reads which aligned to it with partial-matching were retained for annotation[4]. Assembly was performed by MEGAHIT[5], which has several desirable properties compared to SPAdes,[6] chiefly improved speed and much lower

memory requirements, at the cost of producing generally fewer ultra-long contigs[5]. Annotation with BLAST, while normally very time-consuming, was accelerated by searching relatively short contigs

Joel built his model with the intention of having it serve as a simple, fast baseline for more advanced models to supersede. Instead, he ended up the best-performing subgroup in the consortium on both speed and accuracy, though his model tended towards false positives. On reflection, this is unsurprising, as the final step in his pipeline is a BLAST search, which is known for issues with false positives[7]. This method is also by its nature blind to deletions, and performs poorly on subtler forms of engineering.

Singh Group

Principal investigator: Mona Singh (Princeton University)
Team: Anton Persikov

I was not able to get in contact with Mona Singh's group at Princeton, so all information about their approach is secondhand. Reportedly, Anton Persikov, at the Singh group, fared better on avoiding false positives, likely in part to extremely careful data curation. In contrast, Joel initially tried using isolates on GenBank to augment the RefSeq database, but ultimately gave up on sorting the good sequences from the bad, and used base RefSeq. Anton spent significantly more time on data curation for the final pipeline, but I was not able to get in contact with him and cannot report on the details. Anton's method was based on hidden Markov models and **m**aximum **u**nique **m**atches (MUMmer)[8] and generally performed well, but required assembly. It was therefore strictly limited by the quality of the assemblies, which in turn were strictly limited by the depth and quality of the initial sequencing run. They also required substantially longer to process more complex samples, suggesting that it may be a poor fit for the metagenomics use case.

Broad Institute

Principal investigator: Ben Gordon

The core element of Ben's approach was to invert the problem–instead of trying to prove that a read is engineered, prove that it is natural. The goal was to create a "box" product with no human in the loop–software that can be fed raw sequence reads, and spit out a result on the other end.

No new tools were developed, but major engineering effort was put toward integrating dozens of existing bioinformatic tools into software that automatically generates an assembly and annotation workflow and executes it. The approach is agnostic to species, type of engineering, and type of read, but it is quite resource intensive, and takes 12-36 hours to run per genome on a 20-node cluster. It also does not at all address the metagenomics use case; they worked entirely with isolates.

Like most other FELIX performers, Ben encountered issues with data quality, and significant effort was expended on cleaning up RefSeq via metadata filtration. Any sequence deposited by a known synthetic biologist was removed, as well as anything that contained keywords like

"engineered," “artificial,” or "synthetic." However, there were limits to this approach, as the quality of annotations was extremely poor–for example, viruses that infect humans being labeled "human.”

Similar challenges were encountered for establishing engineering markers. Particular markers were readily found for yeast, but bacteria had too many confounders, eg: antimicrobial resistance cassettes, cloning sites, CRISPR. A few additional metrics were added for detecting deletion and compression.

Ben’s software was used, for example, to analyze the SARS-CoV-2 genome for signs of engineering. However, Ben has stated that the work is unlikely to be published, as it primarily represents an achievement in engineering rather than new knowledge. He also appears to have recently left the Broad Institute.

Ginkgo Bioworks - ENDAR

Principal investigator: Joshua Dunn

Ginkgo’s ENDAR (**E**ngineered **N**ucleotide **D**etection **a**nd **R**anking) is conceptually similar to Raytheon’s GUARDIAN in that it is an ensemble approach that integrates information from a variety of modules in order to yield a final judgment. Like most performers, Ginkgo used both read-first (for sensitivity) and assembly-first (for context and interpretability) approaches.

Like the Broad Institute, they focused on clonal samples, and set an internal goal of detecting insertions or deletions of at least 200 base pairs, of determining the organism present. Ultimately, they got detection down to 50 base pairs. However, the limits of detection varied by organism–some organisms (such as *E. coli*) had many confounders, others (such as *S. cerevisae)* had fewer. Many changes were impossible to determine as engineered or not without extremely precise knowledge of what was present. Detection in a metagenomic setting was not a focus, though some work was done on it.

ENDAR features three separate modules:

1. A scan for “smoking gun” signatures of engineering–green fluorescent protein (GFP), antimicrobial resistance cassettes, reporter genes, selection markers, and so on–that are rare in nature. This method is good for finding low-hanging fruit, and certainly desirable to include in any final model, but it is vulnerable to red-teaming. Antimicrobial resistance cassettes are also
2. An alignment-based comparison of raw reads to a large database of natural sequences, with a machine learning module to determine if failed matches were truly engineered or simply previously unseen using compositional patterns in eg: GC content, di- and tri-nucleotide frequency, and k-mer bias. However, reference genomes are not always available, and biological knowledge is required to leverage the compositional approach, which can’t reliably distinguish closely related organisms.
3. A module for reverse-engineering the steps taken to create the sample.

All approaches were combined at a final step with a random forest ensembling method. Performance on clonal samples, though not metagenomics, were “quite good,” though precise performance metrics were not shared. Challenges included difficulties distinguishing engineering from natural transposons, assembly of low-info-content DNA, and spurious signs of engineering resulting from misassembly.

As of June 24th 2022, Ginkgo was the last, or among the last, of performers still involved in FELIX. This suggests that their approach may have been among the best of all FELIX performers. It has been suggested by others involved in FELIX, though not confirmed, that Ginkgo had access to proprietary, high-quality data. If so, given other performers’ difficulty with the use of publicly available data, this is consistent with Ginkgo being an unusually good performer. Ginkgo was selected as the best computational performer to speak at an IARPA press conference**,** where they were noted to perform at 70% accuracy on the final dataset.

Noblis, Inc.

Principal investigator: Sterling Thomas
Team: Lauren Brinkac Leone (Project manager), Tyler Barrus; Bradley Abramson, Granger Sutton (J. Craig Venter Institute, former; Noblis, current); Gary B. Fogel, Enoch. S Liu (Natural Selection, Inc); Todd Michael, Semar Petrus (Salk Institute for Biological Studies)

Of all performers, Noblis was the only one relatively focused on metagenomics, and on computational efficiency in particular. Their work is therefore among the most pertinent for our purposes.

Noblis took an unusual approach in finding samples of engineered sequences–they just got them from Genbank. The thinking was that almost all the *E. coli* strains deposited on Genbank are lab strains, and represent a fairly comprehensive set of possible edits. The real challenge was in obtaining wild-type *E. coli*; Noblis worked with USDA to secure the sequence. This trick might work again with other commonly engineered organisms like *S. cerevisiae* and *B. subtilis,* but probably not too many more.

Ultimately, however, Noblis moved to the generation of in-silico synthetic data, and developed a tool for this purpose. Systematic error rates for sequencing machines were based on the machines owned by Noblis’s three lab locations, and relied on knowledge of conserved organism sequence regions generated by PanOCT[9]. Though this tool is not currently available and there are no explicit plans to open-source it, project lead Sterling Thomas has mentioned being willing to work out some kind of deal for sharing it with us.

BioContext Pipeline

The core of Noblis’s approach is the Biocontext pipeline, an as-yet unpublished model that utilizes previously published methods and tools, including BioVelocity and PanOCT[10].

BioVelocity is a hash-database algorithm conceptually similar to modern-day Kraken[11]. When it was first created, it held a decisive advantage over Kraken, running on an HPC cluster and being written entirely in C++, but over time Kraken has closed most of this gap and is expected

to close it entirely within the coming years. BioVelocity is used in the Noblis FELIX pipeline for initial read-first screening–anything that is on the engineering marker blacklist is flagged, and anything that matches its natural baseline is disregarded. What enables BioVelocity to run on a commercial laptop, however, is the exclusion of core regions of the target genome from its database, since these regions are unlikely to give any information about the presence of engineering (as any edits to those regions would likely result in a nonviable organism)[9].

After filtering for basic blacklists, reads are assembled and passed to PanOCT to compare the full sequence of interest to the existing pan-genome graph. A gradient boosted ensemble algorithm is then used along with BLAST to make a final call about the presence of engineering. Though the Noblis website mentions Bloom filters and count-min sketches as relevant tools for 'reducing the problem set' of engineering detection, these were not included in the final pipeline[12].

This work likely will eventually be published.

Results

Prior to each round of T&E, Noblis conducted an in-house test with their "baseline model"–a human scientist, with whatever tools he wanted available to him. Noblis's pipeline was able to outperform this baseline every time. For the first several rounds of T&E, Sterling described performance as "perfect," with accuracies exceeding 80%. It is unclear to what extent it is possible to compare this performance to other FELIX participants due to the highly variable nature of the T&E samples. In the final round, however, they struggled, and achieved only 60% accuracy on a particularly difficult complex containing several microbes, yeasts, and plants. Analysis of this sample was also extremely time-consuming, requiring nearly a month to run, three weeks of which were spent on assembly alone.

Though Noblis was relatively more interested in metagenomics and had plans to extend their methods to cover that use-case, FELIX leadership were more interested in isolates, and this is ultimately where most of the effort was aimed. The methodology described in the PanOCT repository is specific to isolates, for instance, and the synthetic engineered data tool is designed for isolates only. Reportedly, Sterling's team had a plan for extending the data generation tool to simulating metagenomic reads, but it was never put into development.

Fogel et al (2022)

Noblis is also among the only to have used neural networks (specifically, evolved neural networks) in FELIX-related work, though it doesn't seem to be part of their final pipeline[13]. Fogel et al describe in some detail the process for generating synthetic engineered *Bacillus subtilis* genomes, including various insertions, deletions, and rearrangements, exclusively outside of the determined 'core' regions of the genome. It should be noted that real synthetic biology is quite messy and inefficient, and a study validating the accuracy of the proposed synthetic edits would be of interest.

Quality of alignment of reads to 145 ‘native’ (though likely themselves engineered lab strains) *B. subtilis* genomes were used as input features, which has significant interpretability benefits.

With three-fold cross-validation, average sensitivity and specificity was 100%, albeit on a fairly small dataset of 145 natural genomes and 491 synthetic engineered genomes, and with an artificially selected decision threshold of 0.8. Since no baseline is constructed in this study, it is hard to say if this performance could not be duplicated by a computationally cheaper algorithm.

This result is essentially unsurprising–neural networks trained on an extensively engineered *S. cerevisiae* performed similarly well on other engineered *S. cerevisiae* within the Raytheon BBN group*,* but failed to generalize to other organisms. This approach, too, is limited by the availability of genomes–*B. subtilis* is an extremely well-studied organism, and this will not be the case for the vast majority of what might be found in a wastewater sample.

Wet Lab Approaches

Draper Laboratory

Project lead: Kirsty McFarland
PM: Kevin Remillard
Biosecurity Program Lead: John Julias
Academic collaborator: Mo Khalil (Boston University)

Harvard Medical School:

PI: Mike Springer
PI: Pamela Silver

Raytheon BBN:

Academic collaborator: Eric M. Young (Worcester Polytechnic Institute)

As this review focuses on computational methods of genetic engineering detection; less effort was expended in investigating wet lab approaches and workflows, but for completeness, a summary follows.

Generally all wet lab performers used some variant on targeted detection paired with a standard bioinformatic pipeline. This was Mike Springer’s approach–selecting a limited number of suspicious sequences, such as antimicrobial resistance cassettes and known pathogens like anthrax, and enriching the sample with them, then checking if they were in a surprising context. Pamela Silver took a similar approach, except using biosensors instead of enriched sequencing, though I was not able to get in contact with her for more details. Eric Young developed an integrated workflow for the assembly of engineered yeast genomes[14], notable chiefly for its exhaustiveness.

Finally, Draper Labs, in association with the Khalil group at Boston University, developed two pipelines, TEM-Seq (unknown organism with known modifications) and HIP-Seq (known organism with unknown modifications). HIP-Seq was ultimately not very successful, and was shuttered along some point in development, but TEM-Seq was broadly regarded as a success. TEM-seq worked by using a number of markers of genetic engineering and enriching the sample, using nondestructive capture beads in order to retain contextual information.

Draper was notable for successfully detecting engineering events in the highest level of dilution featured in the FELIX T&E rounds, of one engineered sequence in a million natural ones, including some that IARPA wasn't even aware of. These were likely the remnants of a DNA barcode. Reportedly, these sequences did not appear at all in the unenriched sample.

However, Noblis was able to detect these short contaminants in an unbiased sample. The contaminants may have been present in larger quantities in Noblis's sample, or Draper may have filtered out the shortest contigs at some point along their pipeline, or else this may simply represent the intense variability of any sequencing effort.

Non-FELIX Computational Work

Lawrence Livermore National Laboratory

Much of the LLNL bioinformatics teams' work has been focused on the design of genomic signatures for pathogen defense[15]. Work since 2003 is summarized by Slezak et al in a recent book chapter[16]. Genomic signatures include those for organisms, for functional mechanisms (such as target genes), and those for genetic engineering method signatures. Ideal signatures for organisms must be conserved, reliably detected across all intended organisms, and unique to that organism. All of these methods are again limited by the quality of available data, which is in general poorly-annotated.

Slezak et al outline a few strategies for finding unique and conserved signatures for RNA viruses, a particularly challenging target due to their high mutation rate. The LLNL group solves it with the minimum set clustering (MSC) algorithm. First, any sequences in the target genomes shared by the background genome are removed. Then the remaining unique regions are mined for candidate signatures, to user specifications of GC content, probe length, and so on. The candidate signatures are then clustered by the subset of targets they're to predict. A greedy algorithm can be used to ensure maximum coverage of all possible targets.

The 2020 book chapter speaks limitedly on the subject of engineering, other than to suggest the detection of evidence of engineering and the use of highly multiplexed microarrays containing known signatures of genetic engineering. No meaningful advancements seem to have been made on this front since the 2008 study by Allen et al, discussed below

Allen (2008)

Allen et al apply the alignment-free LLNL method of signature development on the distinguishment of artificial vectors from natural plasmids[17]. A candidate signature is defined as one found in an artificial vector sequence, but not in natural plasmids or in chromosomal DNA–where generally it was more challenging to distinguish artificial vectors from chromosomal DNA than from natural plasmids, owing to the sheer amount of chromosomal DNA.

For any value of k, some candidate signatures will be redundant–exclusively associating the same set of sequences to one another. By grouping these redundant signatures together,

“signature sets” are generated. A greedy algorithm is then used to determine which of these signature sets provide the maximum amount of sequence information. For example–for k=20, there are 1,625,171 signatures, but only 7,270 non-redundant signature sets, and only 364 if selecting for maximum information until total coverage is achieved. Notably, Allen et al find that values below k=18 are likely to produce matches due to random chance. This is consistent with Itoh et al’s investigation of foreign DNA insertion into a much more complex rice genome[18].

The candidate signature sets were evaluated using a BLAST search. As a baseline model, “dummy” signature candidates based only on multiple cloning sites–a hallmark of an engineered plasmid–were created. With five-fold cross-validation, the best baseline model achieved a true positive rate (sensitivity) and true negative rate (specificity) of 92%, while the best k-mer-derived model achieved a sensitivity of 99% and specificity of 99.7%. Removing the signatures responsible for the false positives from the model enabled a sensitivity of 100% while dropping specificity to only 98%.

Like other signature-based methods, this method is vulnerable to red-teaming. However, it has the benefit of ready interpretability, and of efficiency, using only 3 GB of memory and running in linear time.

Since this study was conducted, the number of available artificial vectors has exploded–the Addgene repository now contains over 65,000 deposited plasmids. It may be of interest to repeat this study to include these, along with all newly available natural plasmids.

Wang (2019)

Wang et al represent a more recent attempt at the challenge of distinguishing natural plasmids from artificial vectors[19]. The purpose of this study was primarily to present the *plaster* pipeline, a faster method for the generation of pan-genomes, and was funded in part by another IARPA project, FunGCAT. The primary advantage of this new pipeline is in the ability to quickly update pan-genomes, with the addition of a new genome requiring less than one second for a pan-genome constructed of either 10, 100, or 1000 plasmids.

51,047 complete synthetic plasmid sequences, 73,727 partial synthetic plasmid sequences, and 6,642 natural plasmids were used in the classification task. Pan-genomes were constructed for each dataset. *Plaster* required less than 9 hours to build the complete synthetic pan-genome, less than 35 hours to construct the joint complete-and-partial synthetic pan-genome, and 50.2 hours to construct the natural plasmid pan-genome.

2000 plasmids were withheld from the analysis as a validation set. These were then aligned to the joint synthetic and natural pan-genome with MUMmer. Specificity consistently exceeded 99% for both synthetic and natural plasmids, but sensitivity was poor. Depending on the precise alignment percentage chosen for a decision threshold, sensitivities ranged from 59.7%-77.3% for synthetic plasmids, and 50.2%-76.7% for natural plasmids. Improving the sensitivity for synthetic plasmids necessarily reduced it for natural ones.

Tay (2021)

Tay et al developed INSIDER[20], a tool for alignment-free, agnostic detection of foreign gene content when the organism is poorly characterized or the foreign insert is of unknown origin. Previous work based on the TF-IDF algorithm[10,11] while also aimed at agnostic, alignment-free detection, required some minimal knowledge of the genome's coding regions, while INSIDER requires none.

INSIDER constructs k-mer frequency feature vectors and clusters them. Cluster analysis successfully distinguished yeast, fruit fly, zebrafish, and mammals, though it could not resolve mouse and human genomes, and clustered mitochondrial DNA of different origin together rather than with the species of origin. INSIDER then uses outlier analysis to identify those k-mer vectors (which may represent a read, a contig, or a complete genome) which fall outside the cluster, and flags them. Conceptually, this is similar to the PanOCT pipeline[10]. Interestingly, Tay et al find that 5-mers give the best clustering purity (on typical metrics in unsupervised machine learning), with far decreased cluster quality for 6-, 7- and 8-mers.

This approach was tested on a synthetic dataset of *Saccharomyces cerevisiae* spiked with a CRISPR-Cas9 gene. In the synthetic yeast sample, INSIDER flagged nine outliers. One was the culprit CRISPR gene; another two were known mobile genetic elements. The rest, however, were endogenous yeast genes, and the most significant outlier was a mitochondrial gene.

Though INSIDER is the only truly agnostic, reference-free, alignment-free genetic engineering detection method I am aware of, it has several key limitations.

First is that the obtained results are only possible with assembled contigs, not raw reads. Raw reads alone were insufficient to resolve any information, limiting any analysis using INSIDER by the quality of the assembly. Like other assembly-first methods of detection, there is a risk that errors in assembly may flag contigs spuriously. Secondly, if used naively, this method suffers from low specificity, returning many more false than true positives. Finally, it is very unclear how INSIDER would perform on a much messier metagenomic sample.

Finally, one has to question the utility of a clustering-based method like INSIDER in production compared to more traditional bioinformatic methods already highly optimized for performance. For example, building a pan-genome or Kraken database[11] on the natural genome, and flagging contigs which fail to align above some threshold.

Ultimately, without a shared test dataset, it is difficult to benchmark INSIDER's performance on either speed, scalability, or accuracy.

Discussion

Initially, genetic engineering detection was conceived by us as a machine learning problem, analogous to the genetic engineering attribution problem[23]. At least proximally, this does not

seem to be the case. Genetic engineering detection is a bioinformatics problem that at later stages *might* leverage machine learning techniques.

Before anything else, we must fully define the problem we are trying to solve. We have identified some parameters–we are interested in agnostic methods, in wastewater shotgun sequence data, and we want to be able to process a billion reads a day at reasonable computational cost. But what, exactly, do we want to detect? Which organisms? What kinds of engineering? How "agnostic" is "agnostic"?

Forms of engineering

Signatures

In some senses, signature-based detection is a solved problem.

If we want to catch only the *most* obvious forms of engineering, screening against a Kraken database built on Univec will suffice, followed by assembly of only those reads that pass the screen with MEGAHIT[24]. Methods for the distinction of artificial and natural plasmids exist [17,19], and can likely be improved–one can imagine throwing all manner of neural network architectures at this specific problem.

It is worth noting that Addgene is a biased dataset, containing only those plasmids which researchers have chosen to share, and that any classifier at this stage should be tuned to high sensitivity, low specificity–only classifying sequences as natural if the probability is extremely high. These classification methods, while insufficient on their own, can usefully function as an additional screening step prior to a BLAST search. Annotation with BLAST has its own issues with false positives, as it was never designed with pathogen surveillance in mind,[7] but licensed tools like FAST-NA are a possible alternative.

The problem with this kind of detection method is the problem with signature-based methods in general–they are easy to counter[25]. Nevertheless, it's worth doing, and will probably be most useful in the case of accidental release.

Foreign gene content

We might instead search for 'obvious' engineering that leaves no particular signature, but does integrate foreign gene content into a genome. This case may be particularly relevant to the detection of engineered viruses. At this level, there is no avoiding the necessity of some kind of reference genome. Methods like INSIDER and the PanOCT pipeline, or even just a Kraken database of known natural k-mers to check against, rely on some known ground truth to create clusters, pan-genomes, or databases. For any engineered organism containing foreign gene content we might hope to catch, we must have a reasonably good idea of what a natural version of it looks like.

Some classes of foreign gene content insertions might be obvious enough to be detectable by a hidden Markov model, or another sequential model. This may have some hope of working–after

all, a subtly optimized *E. coli* lab strain might be technically ‘engineered’, but it is likely more similar to a naturally evolved wild-type *E. coli* than an *E. coli* with a large obvious stretch of foreign gene content stuck into it. But some long insertions might be too subtle. One example from FELIX T&E was a *P. aeruginosa* with an inserted gene from closely-related *P. putida.* It is unclear that methods capable of catching obviously-foreign inserts would catch modifications like this.

In general, we should expect this to be easier for viruses (and perhaps yeasts) than for bacteria. Signals of engineering in bacteria are heavily confounded by the natural constructs that exist in bacteria and in some cases may be completely irresolvable.

An option that may work well for long reads is in window-based taxon classification by CDKAM[26]. CDKAM can quickly and efficiently classify reads with a high (10-20%) error rate into taxa. By taking a sliding window along a nanopore long read, (sufficiently) foreign gene content can be readily detected. CDKAM is fast and can certainly work in real-time, but will be limited in sensitivity. We should not expect this strategy to work for an *E. coli* modified with a bacterial plasmid from another gut bacteria, for example.

All else

For subtler forms of engineering–ranging from codon optimization, very small indels or base replacements, to directed evolution–reliance on a reference genome increases. This is problematic–the quality of publicly available data is low, and would require substantial curation before use. In many cases, it is unavoidable to integrate expert biological knowledge.

Functional annotation may be another approach to statistically subtle instances of genetic engineering detection that are nevertheless functionally distinct from the reference[27,28].

Challenges

> *“The current flood of metagenomic data is presenting us with an even larger problem: what exactly do concepts such as ‘species’ and ‘strain’ mean if it turns out that microbial life is a broad spectrum with few well-defined transitions?”*[16]

Reliance on references is especially problematic for the wastewater setting, where most of what is sequenced is previously unknown microbial life. For example–bacteriophages represent the overwhelming majority of viruses found in wastewater[29], and the majority of these are highly fragmented and completely novel[30]. Yet significantly less than half the viral sequences deposited on Genbank are phages. Detecting an engineered phage is likely to be extremely challenging–but how important is it? From a security perspective, leaving any avenue of attack unmonitored is inadvisable, but it may or may not actually be feasible.

There is also the challenge of metagenomic assembly. Current assembly algorithms were never designed with large amounts of data in mind, though this is changing[31,32]. Without a reference, de novo assembly is required, and this is computationally costly as well as error-prone, and assembly errors can resemble engineering[33]. This is particularly the case for viruses–especially

RNA viruses, with their extremely high mutation rates and significant intra-population variation[34]. Beyond the basic problem of increased noise, these properties make viral metagenomic assembly extra challenging[30,35].

One way to mitigate this is to increase sequencing depth and therefore cost, though this cost is expected to fall. But even very thoroughly sequenced samples may fail to assemble accurately, especially highly variable viral species.

Without reasonably high-quality metagenomic assemblies, we are more or less limited to read-focused and therefore largely signature-based methods, and this is especially problematic for most engineered viruses, most of which would not contain any signatures of engineering at all beyond foreign gene content.

Though most assembly algorithms were not designed for today's glut of sequencing data, this is rapidly improving. MetaFlye, for example, is an assembler based on repeat graphs rather than de Bruijn graphs, resulting in more efficient, higher quality assemblies from error-prone long reads[31]. Advances are also being made in the assembly of viruses from metagenomic samples with tools like MetaviralSPAdes and viralFlye[36,37]. Several FELIX teams found that hybrid assemblers of both short and long reads worked best; there is progress on this front as well[38].

Next steps

What is clear is that there are tradeoffs for any form of genetic engineering detection. Fully agnostic detection is possible only with high-quality natural references, is limited by the quality of the assembly; and will not be particularly fast. Fast, sensitive detection systems that can easily handle a billion reads a day almost inevitably rely on signatures.

For this reason, it is imperative to clarify our priorities around detection. There are then a number of additional bottlenecks to address.

Testing & evaluation dataset

A top priority is a *high-quality test dataset*, representing a comprehensive suite of types of engineering that we most want to be able to catch, in more or less the form we expect to find them in. Without one–and ideally, several–it is difficult bordering on impossible to compare different GED methods in a principled way.

It may be possible to generate such a dataset in silico. Peter Ge's thesis includes an appendix from which the Phase 1 T&E sample for Raytheon BBN can be recovered, but these are all isolate samples[4]. Fogel et al[13] describe a process for generating a synthetic engineered dataset for *B. subtilis,* and there are publicly available tools for the generation of metagenomic data[39]. The Noblis FELIX team may be willing to share the software they have developed for this purpose; they also had a plan for extending this work to metagenomics that was never put into practice, but may be in the future.

However, it is unclear to me exactly how trustworthy a synthetic dataset like this would be, even with expert knowledge. It would be preferable to validate this approach with an experimental sample, and in the case of any synthetic metagenome, probably required.

Any such dataset generated in-silico would also be vulnerable to the pollution of databases by engineered sequences. Past work on developing machine learning datasets for engineered sequences would be unlikely to work as intended for this reason[40].

Experimental validation of any proposed synthetic data generation pipeline is particularly important for any viral training data. For obvious reasons, the ability to detect edited pathogenic viruses is particularly important, and for equally obvious reasons, we should not be engineering real pathogenic viruses in the lab.

It's possible that the best course of action is simply to contract the creation of this dataset to a private entity–perhaps Noblis or Ginkgo Bioworks–though there are possible security concerns to be addressed.

Curation of natural data

Another major priority is a curated database of confirmed-natural sequences. Clear priorities are required to begin. For our purposes, what should be considered 'engineered'? If optimized lab strains are to be considered engineered, then nearly every *E. coli* deposited on GenBank is engineered, and it is challenging to verify any publicly deposited *E. coli* as legitimately natural.

RefSeq reference genomes, though curated, are not curated for naturalness; to be deposited in RefSeq, a sequence need only be unique, and complete. If the first deposited sequence happened to be engineered, then that's the reference genome. There's also the matter of metadata–it is untrustworthy, and it is unclear to me to what extent metadata is useful at all.

Screening all of GenBank against a database of known synthetic constructs and well-known engineering markers would be a reasonable (but not sufficient) first step.

There may be other ways to acquire guaranteed-natural reference sequences. The Noblis FELIX team, for example, contacted the USDA for a wild-type *E. coli.* It may also be possible that organisms derived from environmental samples are more likely to be genuinely natural, though the recovery of any single complete genome is unlikely. However, it may still be possible to utilize partial contigs as a reference–perhaps through the construction of a pan-genome.

For this reason, domain expertise is desirable and perhaps required for this task. As above, it may also be possible to contract this out.

A better way to express engineering

In at least one conversation, it has been suggested to me that a major limitation for a GED system is the expressiveness of a sequence alone. For example, while Addgene may be the most comprehensive database available of engineered plasmids, it lacks contextual information

about the overall design of the engineered construct. Detecting a single plasmid might not be very meaningful–but detecting three specific plasmids all in one organism, near other specific genes, may be very meaningful. Pushing state-of-the-art GED systems to greater capabilities may require a way of expressing relationships between genes more concisely, perhaps through something like the Synthetic Biology Open Language standard, but brought to a more complete and comprehensive form. A formal grammar for synthetic biology might, for example, be used to programmatically generate "biologically correct" engineered training data in a way that is currently not possible.

Potential dual use concerns with such a project should be noted, since standardizing the language of synthetic biology would make it both easier to detect, and easier to do.

Surveillance site baseline characterization

For all but the most obvious forms of GED, a known baseline at the surveillance site is imperative. We cannot determine what is present and anomalous if we don't know what's present normally–what is present in the community, what mutational drifts are typical, how quickly a community changes, etc. It is also necessary for the benchmarking of any synthetic datasets and establishing the limits of detection.

There is also the question of whether apparently engineered sequences appear in wastewater samples. It is desirable to establish a reasonable baseline expectation for false positives (or unexpected real positives). Targeted signature-based detection would be a reasonable choice to explore this. Either contracting or collaborating with a FELIX performer focused on signature-based experimental methods may be a reasonable choice in this regard as well.

Potential projects

There are a number of immediately-doable GED projects, suitable for anyone experienced or interested in bioinformatics, metagenomics, synthetic biology, microbiology, virology. Several can be accomplished without significant computational resources. By request, they have been moved out to a separate document, but please just go ahead and request access.

Limitations and Acknowledgements

This document does not include a comprehensive review of another relevant IARPA project, Fun GCAT, though at least one Fun GCAT related project appeared directly relevant for GED[19]. I spoke with project conceptualizer John Julias and took a cursory glance through Fun GCAT-related publications, but otherwise did not investigate deeply or speak with performers.

I also did not include discussions around potentially relevant related computational bioinformatic problems, such as genetic engineering attribution[23,44], detection of horizontal gene transfer events, detection of transposons[45], detection of antimicrobial resistance in wastewater[33], and detection of viral integration in human genomes[47].

I discussed with Jade Zaslavsky particular machine learning methods relevant to GED. This technical discussion is not included, but methods investigated included latent space conditioning for outlier analysis[48], Hopfield networks[49], the HSIC bottleneck[50], DNABert[51], and the ELECTRA discriminator[52].

I was not able to conduct all the FELIX interviews that I had hoped. I have been put in touch with for a tentative meeting with Susan Celnicker (LBNL, T&E), and Lauren Brinkac Leone (Noblis), but have not yet heard back from them. I have contacted but not received responses from Pamela Silver (HMS), Anton Persikov (Princeton), Mona Singh (Princeton), J. Ben Brown (LBNL, T&E), or Tom Slezak (LLNL).

Of course, a major limitation of this document is my own limited background in bioinformatics, metagenomics, or synthetic biology. There are surely many blindspots in the discussion for this reason. Thanks to Jasper Götting, Noga Aharony, Nicole Wheeler, Mike McLaren, and Kyle Bittinger for their expertise and time in helping me contextualize this work. Thanks also to David Markowitz, Mike Springer, Ben Gordon, Peter Ge, Eric M. Young, Nicholas Roehner, Joshua Dunn, John Julias, Sterling Thomas, and Joel S. Bader for their generosity with their time, expertise, and models.

This document may be updated in the future to reflect additional information.

Supplemental Info

This section is included as a resource for anyone interested in pursuing the proximal projects linked above.

Data sources

1. Virus data:
    a. A curated list of accessions for the complete genomes of vertebrate viruses (last updated 2017) and separately, phages
    b. An interactive web-based portal for searching viral reference sequences
    c. RefSeq viral index
    d. Library of engineered adenovirus reads[53]
2. Metagenomic reads:
    a. Wastewater datasets with associated publications
    b. Simulated virome [paper]
        i. Simulated reads, with source genomes
        ii. Two mock viral communities
    c. Metagenome data from a global virome surveillance study
3. Natural plasmids:
    a. A curated database of natural plasmids
    b. ICEberg
    c. PlasmidFinder database (comes with software)

    d. Bacterial virulence factors
4. Synthetic constructs:
    a. NCBI database of synthetic constructs
        i. Viruses
        ii. Phages
        iii. Bacteria
        iv. Vectors
    b. Univec
    c. SEVA (Standard European Vector Architecture)
    d. iGEM Parts
    e. SBOL3 Test Suite
    f. Addgene (available with license)
        i. *E. coli* host strains, that appear in Addgene
            1. Dh5alpha
            2. Top10
            3. XL1 blue
            4. DH10B
            5. ccdB Survival (Top10 derivative)
            6. NEB stable
            7. Stbl3

Software Directory

1. Reference-free analysis tools for metagenomic data:
    a. FastQC - [Software Download] - For quickly getting a look at the general shape of the data, and for quality control
    b. Metaphlan - [Paper] [GitHub] - For profiling the composition of microbial communities with MGS data
    c. Digital normalization technique - [Paper] [GitHub] - Decreases sampling variation, discards redundant data, removes majority of errors - Substantially reduces size of shotgun data, decreases memory + time requirements
    d. SCRuB - [GitHub] - A decontamination tool for microbial community data. May also be used as a batch corrector.

2. Classification tools:
    a. Kraken and friends (2014-2022) - [Website] - ultrafast metagenomic classification/binning
    b. CDKAM (2020) - [Paper] [GitHub] - fast taxon classification by k-mer statistics - like Kraken, but is suitable for long reads
    c. SeqScreen (2022) - [Paper] [GitLab] - screening for sequences of concern in short oligomers

3. Assemblers:

    a. Flye/metaFlye: [Paper] [GitHub] - Assembly with repeat graphs, more suitable for longer, more error-prone reads, better suited for metagenomic assembly/conditions where contaminants are present
    b. MaSuRCA: [Paper] [GitHub] - arguably better than MetaFlye at low-level accuracy (though worse at overall scaffold construction); can do hybrid assembly
    c. MEGAHIT (2015) - [Paper] [GitHub] - the best, fast low-memory-footprint short-read assembler I'm aware of
    d. SPAdes/metaSPAdes (2012-2022) - [Paper] [GitHub] - the best high-memory-footprint short-read assembler I'm aware of
    e. MetaviralSPAdes (2020) - [Paper] [Github 1 2 3] - a metagenomic viral assembler
    f. viralFlye - [Paper] [GitHub] - a metagenomic viral assembler for long reads
    g. DNAStar [Website] [YouTube] - for viral assembly with a GUI
    h. SCAPP (2021) - [Paper] [GitHub] - for plasmid assembly

4. Feature generation tools:
    a. FastK (2021): [GitHub] - A fast k-mer counter
    b. Gerbil (2017): [Paper] [GitHub] - Another fast k-mer counter
    c. GCGC Kmer Tokenizer (2020) - [Website] - A k-mer tokenizer
    d. OligoSearcher (2017): [Paper] [Website] - For generating spectral k-mer features.
    e. Nubeam (2020): [Paper] [GitHub] - For generating short, unique feature vectors

5. Synthetic data generation tools:
    a. ART (2012): [Paper] [Software]
    b. Artificial Fastq Generator (2017): [Paper] [GitHub] - for simulating Illumina reads from genomes
    c. Codon Wizard (2018): [Paper] [Code] - for simulating codon optimized sequences
    d. CAMISIM (2019): [Paper] [GitHub] - for simulating metagenomic reads; can do both Illumina and Nanopore reads
    e. Cello (2016-2022): [Paper v1] [Paper v2] [CAD v1] [CAD v2] [GitHub v1] - tool for design of genetic circuits
    f. GenoCAD (2010): [Paper] [Software Download] - similar to Cello, relatively focused on plant species
    g. SBOL3 (2020): [Paper] [GitHub] - Synthetic Biology Open Language - a standard for specifying synthetic biologic constructs

References

1. Grohmann, L. *et al.* Detection and Identification of Genome Editing in Plants: Challenges and Opportunities. *Front. Plant Sci.* **10**, (2019).
2. Fraiture, M.-A. *et al.* Current and New Approaches in GMO Detection: Challenges and

Solutions. *BioMed Res. Int.* **2015**, 1–22 (2015).

3. Gargis, A. S., Cherney, B., Conley, A. B., McLaughlin, H. P. & Sue, D. Rapid Detection of Genetic Engineering, Structural Variation, and Antimicrobial Resistance Markers in Bacterial Biothreat Pathogens by Nanopore Sequencing. *Sci. Rep.* **9**, 13501 (2019).
4. Ge, Y. Detecting Genetic Engineering with a Knowledge-Rich DNA Sequence Classifier. (Johns Hopkins University, 2020).
5. van der Walt, A. J. *et al.* Assembling metagenomes, one community at a time. *BMC Genomics* **18**, 1–13 (2017).
6. Bankevich, A. *et al.* SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. *J. Comput. Biol.* **19**, 455–477 (2012).
7. Beal, J., Clore, A. & Manthey, J. Studying Pathogens Degrades BLAST-based Pathogen Identification. 2022.07.12.499705 Preprint at https://doi.org/10.1101/2022.07.12.499705 (2022).
8. Marçais, G. *et al.* MUMmer4: A fast and versatile genome alignment system. *PLOS Comput. Biol.* **14**, e1005944 (2018).
9. Sutton, G. *et al.* A pan-genome method to determine core regions of the *Bacillus subtilis* and *Escherichia coli* genomes. Preprint at https://doi.org/10.12688/f1000research.51873.2 (2021).
10. Inman, J. M. *et al.* Large-scale comparative analysis of microbial pan-genomes using PanOCT. *Bioinformatics* **35**, 1049–1050 (2019).
11. Wood, D. E. & Salzberg, S. L. Kraken: ultrafast metagenomic sequence classification using exact alignments. *Genome Biol.* **15**, 1–12 (2014).
12. Elworth, R. A. L. *et al.* To Petabytes and beyond: recent advances in probabilistic and signal processing algorithms and their application to metagenomics. *Nucleic Acids Res.* **48**, 5217–5234 (2020).
13. Fogel, G. B. *et al.* Identification of Synthetic Engineering in Prokaryotic Genomes Using Evolved Neural Networks. in *2022 IEEE Conference on Computational Intelligence in*

Bioinformatics and Computational Biology (CIBCB) 1–8 (2022). doi:10.1109/CIBCB55180.2022.9863024.

14. Collins, J. H. *et al.* Engineered yeast genomes accurately assembled from pure and mixed samples. *Nat. Commun.* **12**, 1485 (2021).

15. Slezak, T. *et al.* Comparative genomics tools applied to bioterrorism defence. *Brief. Bioinform.* **4**, 133–149 (2003).

16. Slezak, T., Hart, B. & Jaing, C. Chapter 20 - Design of genomic signatures for pathogen identification and characterization. in *Microbial Forensics (Third Edition)* (eds. Budowle, B., Schutzer, S. & Morse, S.) 299–312 (Academic Press, 2020). doi:10.1016/B978-0-12-815379-6.00020-9.

17. Allen, J. E., Gardner, S. N. & Slezak, T. R. DNA signatures for detecting genetic engineering in bacteria. *Genome Biol.* **9**, R56 (2008).

18. Itoh, T. *et al.* Foreign DNA detection by high-throughput sequencing to regulate genome-edited agricultural products. *Sci. Rep.* **10**, 4914 (2020).

19. Wang, Q., Elworth, R. A. L., Liu, T. R. & Treangen, T. J. Faster Pan-Genome Construction for Efficient Differentiation of Naturally Occurring and Engineered Plasmids with Plaster. in *19th International Workshop on Algorithms in Bioinformatics (WABI 2019)* (eds. Huber, K. T. & Gusfield, D.) vol. 143 19:1-19:12 (Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019).

20. Tay, A. P., Hosking, B., Hosking, C., Bauer, D. C. & Wilson, L. O. W. INSIDER: alignment-free detection of foreign DNA sequences. *Comput. Struct. Biotechnol. J.* **19**, 3810–3816 (2021).

21. Tsirigos, A. A new computational method for the detection of horizontal gene transfer events. *Nucleic Acids Res.* **33**, 922–933 (2005).

22. Cong, Y., Chan, Y. & Ragan, M. A. A novel alignment-free method for detection of lateral genetic transfer based on TF-IDF. *Sci. Rep.* **6**, 30308 (2016).

23. Crook, O. M. *et al.* Analysis of the first Genetic Engineering Attribution Challenge. *ArXiv211011242 Cs* (2021).

24. Li, D., Liu, C.-M., Luo, R., Sadakane, K. & Lam, T.-W. MEGAHIT: an ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. *Bioinformatics* **31**, 1674–1676 (2015).

25. Mueller, S. On DNA Signatures, Their Dual-Use Potential for GMO Counterfeiting, and a Cyber-Based Security Solution. *Front. Bioeng. Biotechnol.* **7**, (2019).

26. Bui, V.-K. & Wei, C. CDKAM: a taxonomic classification tool using discriminative k-mers and approximate matching strategies. *BMC Bioinformatics* **21**, 1–13 (2020).

27. Balaji, A. *et al.* SeqScreen: accurate and sensitive functional screening of pathogenic sequences via ensemble learning. *Genome Biol.* **23**, 133 (2022).

28. Ng, T. L. *et al.* High-Content Screening and Computational Prediction Reveal Viral Genes That Suppress the Innate Immune Response. *mSystems* **7**, e01466-21 (2022).

29. Nieuwenhuijse, D. F. *et al.* Setting a baseline for global urban virome surveillance in sewage. *Sci. Rep.* **10**, 13748 (2020).

30. Sutton, T. D. S., Clooney, A. G., Ryan, F. J., Ross, R. P. & Hill, C. Choice of assembly software has a critical impact on virome characterisation. *Microbiome* **7**, 1–15 (2019).

31. Kolmogorov, M. *et al.* metaFlye: scalable long-read metagenome assembly using repeat graphs. *Nat. Methods* **17**, 1103–1110 (2020).

32. Zimin, A. V. *et al.* The MaSuRCA genome assembler. *Bioinforma. Oxf. Engl.* **29**, 2669–2677 (2013).

33. Mühr, L. S. A. *et al.* De novo sequence assembly requires bioinformatic checking of chimeric sequences. *PLOS ONE* **15**, e0237455 (2020).

34. Sanjuán, R., Nebot, M. R., Chirico, N., Mansky, L. M. & Belshaw, R. Viral Mutation Rates. *J. Virol.* **84**, 9733–9748 (2010).

35. Smits, S. L. *et al.* Assembly of viral genomes from metagenomes. *Front. Microbiol.* **5**,

(2014).

36. Antipov, D., Raiko, M., Lapidus, A. & Pevzner, P. A. MetaviralSPAdes: assembly of viruses from metagenomic data. *Bioinformatics* **36**, 4126–4129 (2020).

37. Antipov, D., Rayko, M., Kolmogorov, M. & Pevzner, P. A. viralFlye: assembling viruses and identifying their hosts from long-read metagenomics data. *Genome Biol.* **23**, 1–21 (2022).

38. Chen, Z., Erickson, D. L. & Meng, J. Benchmarking hybrid assembly approaches for genomic analyses of bacterial pathogens using Illumina and Oxford Nanopore sequencing. *BMC Genomics* **21**, 1–21 (2020).

39. Fritz, A. *et al.* CAMISIM: simulating metagenomes and microbial communities. *Microbiome* **7**, 1–12 (2019).

40. Painter, C. & Bastian, N. D. Generating genetic engineering linked indicator datasets for machine learning classifier training in biosecurity. in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications III* vol. 11746 517–522 (SPIE, 2021).

41. Herrmann, A.-K. *et al.* A Robust and All-Inclusive Pipeline for Shuffling of Adeno-Associated Viruses. *ACS Synth. Biol.* **8**, 194–206 (2019).

42. Guenther, C. M. *et al.* Synthetic virology: engineering viruses for gene delivery. *WIREs Nanomedicine Nanobiotechnology* **6**, 548–558 (2014).

43. Rehbein, P., Berz, J., Kreisel, P. & Schwalbe, H. “CodonWizard” – An intuitive software tool with graphical user interface for customizable codon optimization in protein expression efforts. *Protein Expr. Purif.* **160**, 84–93 (2019).

44. Wang, Q., Kille, B., Liu, T. R., Elworth, R. A. L. & Treangen, T. J. PlasmidHawk improves lab of origin prediction of engineered plasmids using sequence alignment. *Nat. Commun.* **12**, 1167 (2021).

45. Vermeulen, S. Detecting (novel) transposons in metagenomic data using custom Hidden Markov Models. (2022).

46. Exploration of the antibiotic resistome in a wastewater treatment plant by a nine-year longitudinal metagenomic study. *Environ. Int.* **133**, 105270 (2019).

47. Cantalupo, P. G. & Pipas, J. M. Detecting viral sequences in NGS data. *Curr. Opin. Virol.* **39**, 41–48 (2019).

48. Norlander, E. & Sopasakis, A. Latent space conditioning for improved classification and anomaly detection. *ArXiv191110599 Cs Stat* (2019).

49. Ramsauer, H. *et al.* Hopfield Networks is All You Need. Preprint at https://doi.org/10.48550/arXiv.2008.02217 (2021).

50. Ma, W.-D. K., Lewis, J. P. & Kleijn, W. B. The HSIC Bottleneck: Deep Learning without Back-Propagation. *Proc. AAAI Conf. Artif. Intell.* **34**, 5085–5092 (2020).

51. Ji, Y., Zhou, Z., Liu, H. & Davuluri, R. V. DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. *Bioinformatics* **37**, 2112–2120 (2021).

52. Clark, K., Luong, M.-T., Le, Q. V. & Manning, C. D. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. Preprint at https://doi.org/10.48550/arXiv.2003.10555 (2020).

53. Zhang, W. *et al.* An Engineered Virus Library as a Resource for the Spectrum-wide Exploration of Virus and Vector Diversity. *Cell Rep.* **19**, 1698–1709 (2017).