Dr. Jeanna Matthews, Professor of Computer Science, Clarkson University

Materials for 2022 NACDL & The Samuelson Clinic's Spring Seminar
Unlocking the Black Box: Challenging Surveillance Tools & Technologies in Criminal Cases
Chicago, IL , May 16 & 17, 2022

Litigating DNA Software Panel

Summary: This panel session with Megan Graham, Khasha Attaran and Jeanna Matthews features a discussion about *United States v. Ellis*, the first federal case to order disclosure of the source code to TrueAllele, a program that uses a black box algorithm to analyze DNA evidence. We will talk about what sorts of information you should push for access to in DNA software cases and strategies for litigating access to software source code.

1. Source Code and Beyond, What Type of Materials Are Relevant

United States v. Ellis was the first federal case to order disclosure of the source code to TrueAllele, a program that uses a black box algorithm to analyze DNA evidence, but the scope of the protective order actually included substantially more than source code. Here is a section of the protective order:

> *The protections of this Order cover the following material that is hereby ordered to be disclosed to Defense Counsel and their Experts:*
>
> *a) TrueAllele source code for the version used in the instant case;*
>
> *b) All software dependencies including third-party code libraries, toolboxes, plug- ins, and frameworks;*
>
> *c) Software engineering and development materials describing the development, deployment, and maintenance of the version(s) of the TrueAllele software system used in the instant case, including the software engineering documents recommended by organizations such as the Institute of Electrical and Electronics Engineers or the Internal Organization for Standardization;*
>
> *d) All records of software glitches, crashes, bugs, or errors encountered during the TrueAllele developmental validation study;*
>
> *e) Software version numbers of the components of the TrueAllele system used for the developmental validation study;*
>
> *f) All records of unexpected results, including false inclusions, false exclusions and the conditions under which the unexpected results were achieved.*

When the goal is to assess the reliability of software for a particular case, it is important to request access to as many of the relevant software development materials as possible. This provides essential context for what testing was conducted, what where the criteria used for acceptance testing, what errors have already been found and fixed in the system and more. Industry organizations such as the Institute of Electrical and Electronics Engineers (IEEE) offer standards for verification and validation (V&V) that outline the types of materials that are considered best practices and in fact are required in many critical software industries. Given our

experience in U.S. v. Ellis, I would recommend working with your expert witness team to write an even more specific and detailed list of software dependencies and V&V materials needed for effective review and testing.

2. Problems with the Access We Were Provided

In U.S. v. Ellis, we were given access to 2 computers. On one computer, there was a working black-box executable. We were able to run full tests but without any access to the inner workings of the system. On the other computer, we had access to the source code of True Allele. However, we could not assemble it into a working system for many reasons. First, despite the order specifying all software dependencies, many of the dependencies were missing. Second, some of the source code files were not provided and all the source code files had been removed from their natural organization structure (directories and sub-directories grouping like files together) and put all together in one flat collection. Third, the computer was running an operating system on which True Allele was not normally constructed. Fourth, no build instructions were provided. For all these reasons, we were unable to assemble the provided source code into a working system.

The first computer with a working black-box executable was like a running car for which we could not pop the hood. The second computer with the source code was like a pile of parts – nuts, bolts, pulleys, screws, etc. - that came from a completely dissembled car with some parts missing, no schematics for putting it together, and parts from disparate components mixed together (e.g. all the tiny parts of the transmission, engine and brakes mixed together). One of the prosecutions own experts agreed that they would be unable to identify which functions/parts would have been used in processing Mr. Ellis' data from the information provided to us.

The prosecution argued that it was sufficient for us to test individual functions (e.g. test the strength of an individual screw or pulley) and that it would too much risk to Cybergenetics to allow us sufficient access to build a completely working version of the system that could be examined. They did not explain how there would be any additional IP risk to them from allowing us to build the system vs. reading the provided source code. We argued that there was no additional IP risk but simply an attempt to make it impossible for us to carry out the type of testing we proposed.

The prosecution argued that we could us publicly available data sets (e.g. a data set from Rutgers University) to test the black-box executable. However, there are substantial problems with this assertion. First, there is no definition of the correct LR for any given test making it difficult to truly fail a test. (The range of LRs accepted for known samples is wide and vague.) Second, there is no way to be sure that the executable system was indeed built from exactly the source code provided to us. Matching version numbers alone are an unreliable reflection of whether any other portions of the source code differed. It is completely possible to make a change in source code without changing the version number. Third, it would take substantial engineering effort and run time to prepare data sets like the Rutgers dataset for use and to automate the running of hundreds or thousands of tests. Fourth, there is no way to rule out an error in Mr. Ellis' case using a dataset like the Rutgers set. Any given clients' data could trigger

bugs not seen with other data. We asked to examine the internal operation of the system on Mr. Ellis' specific data, but were prevented from doing so, but the type of access provided.

3. What Types of Access Should You Push For

It is important to note that source code access is more frequently provided to look violations of IP rights (e.g. code that has been copied from a company by a competitor). Looking for signs of copied code is a completely different kind of review than assessing software for reliability/investigating whether the software is behaving in an erroneous manner on a particular client's data.

It will greatly impede review if source code is not provided in a searchable digital format. When reading software, it is important to be able to jump from the call site of a function to its implementation. There are standards tools like Integrated Development Environments (IDEs) that are essential to doing this quicky and efficiently. As much as possible. You should ask that experts be provided a working environment like the development team has access to with all the tools, editors, IDEs, etc. needed to efficiently read, follow and review the code.

In order to identify what code is actually executed on the data at issue in a particular case, it would be necessary to be able to run through the execution of the software in tool such as a debugger that follows execution step by step through the code. This is a commonly-used best-practice tool and I have heard no argument for why it increase the risk to a company's IP to enable experts to examine code with such a tool vs. just reading the source code.

It is also important to ask for access to all software dependencies including external databases that provide input to the execution of the software. This is essential because errors can just as easily be found in databased on which the software relies as directly in the source code. To illustrate why this is so, consider the example of software that recommends a restaurant to you based on your location. If it incorrectly recommends a restaurant hundreds of miles away after consulting a database of possible restaurants, then the problem causing the error may very well be in the contents of that database. If the expert is prevented from examining important dependencies like databases, then it may be impossible for them to identify an error impacting a particular recommendation, especially when unlike with far away restaurants the user is unable to recognize an incorrect output. This is the case with outputs like LRs for which there is no ground truth that would allow an operator to recognize and call attention to a suspicious result.

It is worth noting that access to external databases could be handled securely in a number of ways. First, if as in Ellis, a fully working black-box executable is also provided with access to offsite databases there should be no additional security concerns. Second, connections to external database could be secured using encryption technology like VPNs. Third, a copy of the database could be run on the same machine or on another machine located on the premises where the expert is examining the system.

It is worth noting that for many software systems, especially complex systems like probabilistic genotyping, experts from multiple disciplines may need to be involved in the review (e.g. computer scientists, statisticians, geneticists, etc.). If possible, would highly recommend to structure the review in such a way that expert witnesses can collaborate without all having to travel to the same location at the same time.

Finally, it is important to consider what tools experts are allowed to bring in with than and what notes experts are allowed to take to document their findings for communication to the court. It can be important to bring in data sets and tools. The prosecution in U.S. v. Ellis even suggested using the Rutgers dataset which would have required a sizable set of data and tools to be brought in. I have mentioned other problems with the specific suggestion but the fact that the prosecution suggested illustrates the types of testing and access that could be expected/needed. Consider mentioning in the protective order the ability to bring in tools and datasets.

It is also important to consider requesting that the expert be able to take a limited set of digital notes e.g. recording small relevant portions of code and other details for comment and inclusion in their report to the court.

4. Other Helpful Resources

***Mats Heimdahl and Jeanna Matthews*,**
Amici Curiae Brief in New Jersey v. Pickett , October 14 2020. Full Decision , Some key quotes
Declaration in US v. Ellis, Heimdahl and Matthews, November 16 2020. Response, Nathan Adams and Jeanna Matthews, February 16 2021. Decision , Some key quotes

S. Lacambra, J. Matthews and K. Walsh.
Opening the Black Box: Defendants' Rights to Confront Forensic Software
The Champion , May 2018.
PDF

J. Matthews, G. Northup, I. Grasso, S. Lorenz, M. Babaeianjelodar, H. Bashaw, S. Mondal, A. Matthews, M. Njie, J. Goldthwaite
When Trusted Black Boxes Don't Agree: Incentivizing Iterative Improvement and Accountability in Critical Software Systems
Proceedings of the 2020 AAAI/ACM Conference on Artificial Intelligence, Ethics and Society (AIES) , New York, New York, USA, February 7-8 2020.
PDF (Paper), Slides

J. Matthews, N. Adams, J. Goldthwaite
Decoding Probabilistic Genotyping Software,
Questioning Forensics 2020, 22 and You: Fighting for Privacy & Justice in an Age of Genetic Surveillance, Brooklyn Law School, Brooklyn, New York, USA, January 14-25 2020.

N. Adams, S. Lorenz, M. Babaeianjelodar, J. Matthews, D. Krane
Quantifying the impact of post-validation modifications to Forensic Statistical Tool Criminalistics Track, American Academy of Forensic Sciences (AAFS) 2019 Annual

Scientific Meeting , February 18-23 2019.
Abstract: PDF

J. Matthews, S. Lorenz, M. Babaeianjelodar, A. Matthews, M. Njie, N. Adams, D. Krane, J. Goldthwaite, C. Hughes
The Right To Confront Your Accusers: Opening the Black Box of Forensic DNA Software
Proceedings of the 2019 AAAI/ACM Conference on Artificial Intelligence, Ethics and Society (AIES) , Honolulu, Hawaii, January 27-28 2019.
PDF

Brown Institute Magic Grant,
Decoding Differences in DNA Forensic Software , 2018-2019.
Magic Grant Profile
Example of impact: STRmix ruled inadmissable in US v. Gissantaner, October 16 2019. (PDF)

J. Matthews, N. Adams, J. Greco,
You're just complaining because you're guilty: A DEF CON Guide to Adversarial Testing of Software Used In the Criminal Justice System
DEF CON 26 , Las Vegas, August 9-12 2018.
Schedule , Slides , Video

J. Matthews, N. Adams, J. Greco,
You're just complaining because you're guilty: A Guide for Citizens and Hackers to Adversarial Testing of Software Used In the Criminal Justice System
Bsides Las Vegas 2018, Las Vegas, August 7-8 2018.
Video

Michael Edge, Jeanna Matthews,
Open practices in our science and our courtrooms ,
Open Science Framework, July 14 2021.