Hi, I’m a Professor McDougal, and Professor Wayne is also in the back. If you haven’t signed in, please make sure that you pass this, get a chance to sign the sign in sheet. So today we are very, very privileged to be joined by Professor Naim Rashid from the University of North Carolina Chapel Hill, Professor Rashid got his bachelor’s in biology from Duke, and his PhD in biostatistics from UNC Chapel Hill. He’s the author of 34 publications, and he holds a patent on methods in composition for prognostic and/or diagnostic supply chain of pancreatic cancer. He’s currently an associate professor at UNC Chapel Hill’s department of biostatistics, and he’s also affiliated with their comprehensive cancer center there. With that, Professor Rashid, would you like to take it away?

Sure.

It looks like it says host disabled screen sharing. Chuckling.

All right, give me one second.

Thank you.

I’m trying to do.

(indistinct)

Okay, you should be, you should be able to come on now.

All right.

Can you guys see my screen?

All right.
Can you guys see this?
There we go.
Perfect. Thank you.
Okay, great.
So yes, thanks to the department for inviting me to speak
today, and also thanks to Robert and Wayne for organizing.
And today I’ll be talking about issues regarding replicability in general,
and about specific issues in genomics itself, and then I’ll be moving on to talking
about a method that we’ve proposed to assist with issues regarding data integration, and learning
in this environment when you have a heterogeneous data sets.
I’ll talk a little bit about a case study
where we apply these practices to subtyping pancreatic cancer, touch on some current work
that we’re doing, and then end with some concluding thoughts.
And feel free to interrupt, you know,
as the talk is long, if you have any questions.
So I’m now an associate professor in the department
of biostatistics at UNC.
My work generally involves problems surrounding cancer and genomics, and more recently
we’ve been doing work regarding epigenomics.
We just recently published a supply-connected package called Epigram for a consistence of differential key calling, and we’ve also done some work in model-based clustering.

We published a package called, FSCSeq, which helps you derive and discover clusters from RNA seq data, while also determining clusters in specific genes.

And today we’ll be talking more about the topic of multi-study replicability, which is the topic of a paper that we published a year or two ago, implementing some of the methods.

So before I get deeper into the talk, one of the things I wanted to establish is this definition of what we mean by replicability. You might’ve heard the term reproducibility as well, where reproducibility is the ability to take coding data from a publication, and to rerun the code and get the same results as the original publication.

Where replicability, we’re defining as the ability to be run an experiment generating new data, and get results that are quote, unquote “consistent” with that of the original study.

So in this sort of context, when it comes to replicability, you might’ve heard about publications that have come out
In the past, there have been issues regarding replicating research published in the scientific literature. One paper in PLOS Medicine was published in 2005, and since then, there have been numerous publications discussing problems regarding replicability and ways to address it. The problem has grown large enough to have its own Wikipedia entry discussing the crisis, with a long list of examples about issues in replicating results from scientific studies.

This issue has been known for some time and also applies to situations where, for example, developing clinical prediction models in genomics. Let’s consider an example: we wanted to develop a model that predicts some clinical outcome $Y$, given a set of gene expression values $X$, in the population of metastatic breast cancer patients. We wanted to develop a model that predicts $Y$, given $X$, in the population of metastatic breast cancer patients. We wanted to develop a model that predicts $Y$, given $X$, in the population of metastatic breast cancer patients.

Now, if we have data from one particular trial that pertains to this patient population, we can use it for clinical decision-making. And so the purpose of this sort of exercise is to hopefully translate this sort of model to the clinic.
in addition to having gene expression data,
let’s say that we derived a model, let’s say that we’re modeling some sort of binary outcome,
and in this model, we used a cost report, or penalized logistic regression model
that we fit to the data to try and predict the outcome, given the gene expression values.
And here we obtained, let’s say, 12 genes
after the fitting process, and the internal model 1 UNC on the sort of training subjects is 0.9.
But then let’s say there’s another group at Duke that’s using data from their clinical trial,
and they have a larger sample size.
They also found more genes, 65 genes,
but have a slightly lower training at UNC.
However, we really need to use external validation to sort of get an independent assessment of how well each one of these alternative models are doing.
So let’s say we have data from a similar study from Harvard,
and we applied both these train models to the genomic data from that study at Harvard.
We have the outcome information for those patients as well,
so we can calculate how well the model predicts on those validation subjects.
And we find here in this data set,
model 2 seems to be doing better than model 1,
but if you try this again with another data set from Michigan, you might find that model 1 is doing
better, better than model 2.

So the problem here is where we have researchers that are pointing fingers at each other, and it's really hard to know, “Well, who's who's right?” And why is this even happening in the first place, in terms of why do we get different genes, numbers of genes,

0:07:08.797 and each of the models derived from study 1 and study 2?

And why are we seeing very low performance in some of these validation datasets? So here's an example from 2014, in the context of ovarian cancer.

The authors basically collected 10 studies, all were microarray studies. The goal here was to predict overall survival in this population of ovarian cancer patients, given gene expression measurements from this microarray platform.

So through a series of really complicated cross-fertilization approaches, the data was normalized, and harmonized across the studies, using a combination of ComBat and frozen RNA, and then they took 14 published prediction models in the literature, and they applied each of those models to each of the subjects from these 10 studies, and then compared the model predictions across each subject. So each column here in this matrix is a patient, and each row is a different prediction model, and each cell represents the prediction.
from that model on that patient.

So an ideal scenario, where we have the models generalizing and replicating across each of these individuals, we would expect to see the same color value, meaning that the predictions are consistent. But clearly we see here that the predictions are actually very inconsistent, and very different from each other.

In addition, if you look at the individual risk prediction models that the authors used, there was also substantial differences in the genes that were selected in each of these models. So there’s a max 2% overlap in terms of common genes between each of these approaches. And one thing to mention here is that each one of these risk-prediction models were derived from separate individual studies. So the question here is, if you were a clinician, you’re eager to sort of take the results that you’re seeing here, and extend to the clinic, which model do you use, which is right? Why are you seeing this level of variability? This is, of course, concerning, if you, if your goal is to move things towards the clinic, and this also has implications in terms of, you know, getting in the way of trying to approve the use of some of these, and for clinical use.
So why is this happening?

So there’s been a lot of studies have been done that have tied issues to, obviously, sample size in the training studies, smaller sample sizes, and models trained on them may lead to more unstable models, or less accurate models.

Between different studies, you might have different prevalences of the clinical outcome. In some studies, you might have higher levels of response, and other studies, you might have lower levels of response.

For example, if you have this binary clinical outcome, and also there’s issues regarding differences in lab conditions, where the genomic data was extracted. We’ve seen at Lineberger that, depending on the type of extraction, RNA extraction kit that you use, you might see differences in the expression of a gene, even from the same original tumor.

And also the issue of batch placement, which has been widely talked about in the literature, where depending on the day you run the experiment, or the technician who’s handling the data, you might see slight differences.

There’s also differences due to protocols. Some trials might have different inclusion and exclusion criteria, so they might be recruiting a slightly different patient population, even though they might be all...
in the context of metastatic breast cancer.
All of these things can help impart heterogeneity between what the genomic data and the outcome data across different studies.
In the context of genomic data in particular, there’s also this aspect of data preprocessing.
For the normalization taking that you use is very important, and we’ll talk about that in a little bit.
And it’s a very critical part when it comes to training models, and trying to validate your model on other datasets, and depending on the type of normalization you use, this could also impact how well your model works.
In addition, there’s also differences in the potential way in which you measure gene expression.
Some trials might use an older technology called microarray.
I know other trials might use something relatively more recent called RNAC,
or a particular trial might use a more targeted platform like NanoString.
So the differences in platform also can lead to differences in your ability to help validate some of these studies.
If you train something in marker rate, it’s very difficult to take that model, and apply it to RNAC, because the expression values are just are just different.
And so, as I mentioned before, this also impacts through to normalization on model performance as well.
So the main thing to remember here is that the traditional way in which prediction models, based on genomic data for using the clinical training is typically on the results from a single study.

To talk a little bit more about question of between-study normalization, and the purpose of this is to put the expression data on basically an even scale, which helps facilitate training. If there's global shifts, and some of the expression values in one sample versus another, it's very difficult to train an accurate model in that particular scenario.

So normalization helps to align the expression you get from different samples, and hopefully across the between difference as well. And so the goal here is to eventually predict this outcome in a new patient, you plug in the genomic data from a new patient in order to get the predicted outcome for that patient based on that training model.

So the, in order to do that, you also have to normalize the new data to the training data, right? Because you also want to put the new data on the same scale as a training data, and in the ideal scenario, you would want to make sure that the training samples you use to train your original model are untouched, because what some people try to do is they try to sort of sidestep this normalization issue, they would combine the new data with the old training data,
and renormalize everything at once.

And the problem with this is that this changes your training sample values, and in a sense, would necessitate the fact that you need to retrain your old model again.

And this leads to instability, and lack of stability over time in terms of the model itself.

So in the prior example from ovarian cancer, this is not as big of an issue, because you have all the data you want to work with in hand.

This is a retrospective study, you have 10 data sets, so you just normalize everything at the same time, that’s in ComBat and frozen RNA.

And so you can split up those studies into separate training and test studies, and they’re all rated on the same scale.

But the problem is that in practice, you’re trying to do a prospective type of analysis, where when you train your model, you’re normalizing all of the available studies you have, let’s say, and then you use that to predict the outcome in a future patient, or a future study.

And so the problem with that is that you have to find a good way to align, as I mentioned before, the data from that future study for your training samples, and that may not be an easy task to do, especially for some of the newer platforms like RNAC.

So taking this problem a step further,
that’s available to begin with?

This really is going to make things difficult in terms of the training in the model in the first place.

Another more complicated problem is that you might have different types of platforms at that training time.

For example, you might have the only type of data that’s available from one study is NanoString in one case, and another study it’s only RNAC, so what do you do?

And looking forward, as platforms change, as technology evolves, you have different ways of measuring gene expression, for example.

So what do you do with the models that are trained on old data, because you can’t apply them to the new data?

So oftentimes you find this situation where you have to retrain new models on these new platforms, and the old models are not able to be applied directly to this new data types.

So that leads to waste here.

If you take all of these problems together, regarding cross-study normalization, and changes in platform, and a lot of the other issues, you know, regarding replicability that I mentioned, it’s no wonder that there’s only a small handful of expression-based clinically applicable assets have been approved by the FDA, like Oncotype DX, MammaPrint,
and Prosigna, because this is a very, very tough problem.

So I want to move on with that, to an approach that we proposed to help tackle this sort of issue by using this idea of multi-study learning, where instead of just using, and deriving, and generating models from individual studies, we combine data from multiple studies together, and create a consensus model that we use for prediction, which will hopefully be more stable, and more accurate down the road.

So this approach of combining data is called horizontal data integration, where we're merging data from let's say K different studies. And the pro of this approach is that we get increased power, and the ability to reach some sort of consensus across these different studies.

The negative is that the effect of a gene and its relationship to outcome may actually vary across studies, and also by, you know, depending on, and also the way that you normalize the genes may also vary across studies too if we're using published data from some prior publication. There's also this issue of sample size and balance.

You might have a study that has 500 subjects, and another one that might have 200 subjects.

So there are some methods that were designed to account for between-study heterogeneity after you do horizontal data integration.
One is called the meta-lasso, another is called
the AW statistic, but these two methods don’t really
have any prediction aspect about them. They’re more about feature selection.
Ensembling is one approach that can directly account
for between-study heterogeneity after horizontal data integration, but there’s
no explicit feature selection step here.
But all of these approaches assume that the data has been pre-normalized.
As we talked about before,
for prospective decision-making, based off a train model,
that might be prohibitive in some cases,
and we need a strategy also to easily predict
and apply these models in new patients.
Okay, so moving on, we’re going to talk first
about this issue of how do we integrate data,
and sort of sidestep this normalization problem
at training time, and also at test time where we,
when we try to predict in new subjects?
So the approach that we put forth is to use
what’s called top scoring pairs, which you can think of
as a rank-based transformation of the original set
of gene expression values from a patient.
So the idea here originally,
when top scoring pairs were introduced,
was you’re trying to find a pair of genes
where it’s such that if the expression of gene A
in the pair is greater than gene B, that would imply
that the, let’s say, the subtype for that individual is, say, subtype one, and if it’s less, then that implies subtype zero with high probability. Now, in this case, this sort of approach was developed with when one has a binary outcome variable.

In this case, we’re talking about subtype, but it could also be tumor response or something else. So essentially what you’re doing is that you’re taking these continuous measurements in terms of gene expression, or integer, and you are converting that, transforming that into basically a binary predictor, which takes on the value of the zero or one. And the hope is that that particular transformed value is going to be associated with this binary outcome.

So the simple assumption in this scenario is that the relative rank of these genes in a given sample is predictive of subtype, and that’s it. And so the example here I have on the right is an example of two genes, GSTP1 and ESR1. And so you can see here that if you’re in the upper left quadrant, this is where this gene is greater than this gene expression, it’s implying the triangle subtype with high probability, the triangle subtype with high probability, and otherwise it implies the circle subtype. So that’s the general idea of what we’re going for here. It’s a sort of a rank-based transformation of the original continuous predictor space. So the nice thing about this approach,
because we’re only based on the simple assumption, right?
That we’re only caring about the relative rank within a subject, this makes this particular new transformed predictor relatively invariant to batch effects, pre-normalization, and it also most importantly, simplifies merging data from different studies.
Everything is now on the same scale, zero to one, so it’s very easy to paste together the data from different studies, and we can sidestep this problem of trying to pick a cross-normalization approach, and then work in this sort of transformed space.
The other nice thing is that this is easily computable for new patients as well.
If you have a new patient that comes into clinic, you just check to see whether the gene A is greater than gene B in terms of expression, and then you have your value for this top scoring pair, and we don’t have to worry as much about normalizing this patient’s raw gene spectrum data to the training sample expression values.
So essentially what we’re doing here is that we’re, let’s enumerate all possible gene pairs for us, instead of a candidate genes, and each column here in this matrix shown on the right pertains to the zero one values for a particular gene pair J.
And so this value takes the value of one, it is greater than B, in sample I, in pair j, and zero otherwise.
And then we merge over the common top scoring pairs.
So in this example have data from four different studies, each indicator by a different color here. In the first track, and this data pertains to data from two different platforms, and three different cancer types. And so the clinical outcome here is binary subtype, which is given by the orange and the blue color here.

So you can see here that we enumerated the TSPs, we merged the data together, and now we have this transformed predictor agents. And the interesting thing is that you can definitely see some patterning here. With any study where you have a particular set of TSPs that had taken a value of one, when the subtype is blue, and it flips when it’s orange.

And we see the same general pattern seem to replicate across different studies, but not every top scoring pair changes the same way across different studies. So if we cluster the rows here, we can also see some patterns sort of persist where we see some clustering by subtype, but also some clustering by study as well. And so what this implies is that there’s a relationship between TSPs and subtypes, and that can vary across studies, which is not too different from what we’ve talked about regarding the issues we’ve seen in replicability in the past. So ideally we would like to see a particular gene pair,
or TSP vector here take on a value of one, only when there’s the orange subtype, and zero in the blue subtype, or vice versa. And we wanted to see this pattern replicated across patients in studies, but we see obviously that that’s not the case. So the question now that we’ve sort of introduced, or proposed is this sort of approach to simplify data merging in normalization. The question now that we’re sort of dealing with is well, how do we actually now find features that are consistent across different studies in their relationship with outcome, and also estimate their study-level effect, and then use them for prediction?

So that leads us to the second part of our paper, where we developed a model to help select these particular study-consistent features while accounting for study-level heterogeneity. So to sort of illustrate the idea behind this, let’s just start with a simple simulation where we’re not doing any normalization, we’re not worrying about resuming, everything’s fine in terms of the expression values, and we’re not doing any selection, no TSP transmission either.

So we’re going to assimilate data pertaining to two, let’s say, known biomarkers that are associated with binary subtype. We’re going to generate K datasets, and we’re going to try three different strategies.
for learning a prediction model two to these data sets.
And at the end, we’re going to validate each of those models on an externally-generated data set to compare their prediction performance.
So to do this, we’re going to fit and assume for each study that we can fit it with a logistic regression model to model by our outcome with these two predictors, and in generating these K data sets, we’re going to vary the number of with respect to K. So we might generate two trained data sets five or 10, and also change the total sample size of each one, and make sure that the sample sizes are in balanced across the different studies, and then assume values for the coefficients for each of these predictors to be these values here, and lastly, to induce some sort of heterogeneity across the different training datasets, we’re gonna add in sort of like a random value drop from the normal distribution, where we’re assuming this level of variance for this value. So basically we’re just injecting heterogeneity into this data generation process. So after we generate the training studies, we’re going to apply three different ways or strategies to the training data. The first is the individual study approach, which we’ve talked about before, where you train a generalized model separately for each study. The second approach is where you merge the data. Again, we’re ignoring the normalization problem here.
in simulation, obviously, and then train a single GLMM for the combined data, and then lastly, we’re going to merge the data, and train a generalized linear mixed model, where we explicitly account for a random intercept, and a random slope for each predictor, assuming, you know, a study-level random effect. So after we do that, we’ll generate a validation dataset from the same approach above, and then predict outcome in this validation dataset with respect to the models derived from each of these three strategies. So if we look at the individual strategy performance, where we fit a GLM logistical regression model separately for each study, and then apply it to this validation data set, we can check the prediction accuracy, we can find that, due to the induced level of heterogeneity between studies in predictor effects, in one study, we do really poorly, and another study we do really well, and this variation is entirely due to variations in the gene subtype relationship. And these predictions obviously vary as a result across the different studies. And this will reflect a little bit of what we see in some of the examples that we showed earlier, studies that were trained on different data sets. And then the second approach is where we combine the data sets, and train a single logistical question model.
to predict outcome. And so we see what the median prediction error is better than most of the models here, but if we fit the GLMM, the median prediction (indistinct) gets better than some of the other approaches here. So this is basically just one example. So we did this over and over a hundred times for every single possible simulation condition, varying K, and the heterogeneity across different studies. And some of the things that we found was that the individual study approach had, as you can see, the worst prediction error overall, combining the data improved this a little bit, but the estimates for the coefficients from the combined GLMM were still biased. There’s supposed to be two in this extreme scenario. And a kind of heterogeneity with the GLMM mixed model had the best performance out of the rest, and also had the lowest bias in terms of the regression coefficients as well. So this is great, but we also have a lot of potential types of pairs. We can’t really estimate them all with a GLMM mixed model, so we need to find a way where we can, at least in reasonable dimension, figure out a way which fixed effects are non-zero, while accounting for, you know, this sort of study-level heterogeneity for each effect. So this led us to develop a pGLMM, which is basically...
A high-dimensional generalized intermixed model, where we are able to select fixed and random effects simultaneously using a penalization framework. So essentially here, we’re assuming that all the predictors in the model, we assume a random effect, a random slope for each one, and so we were aiming to select the features that have non-zero fixed effects in this particular approach, and indeed we’re assuming these are going to be study-consistent. And to do this, we’re going to reorganize the linear predictor from the standard GLMM, so basically we’re starting with the same general likelihood for, you know, the generalized mixed model. Here, Y is our outcome, X is our predictor, alpha is the, alpha K is the random effect for the case study, fi here is typically assumed to be multi, very normal, means zero, and a covariant on some sort of unstructured covariance matrix typically. And so to sort of simplify this, we factor out the random effects covariance matrix, and incorporate into the linear predictor. And with some more reorganizing, now we’re able to select the fixed effects and determine which random effects have true non-covariance, using this sort of joint penalization framework. If you want more detail, you can check out the publication.
0:28:27.54 –> 0:28:31.34 that I linked above, and I also forgot to send out
0:28:31.34 –> 0:28:33.01 the link to this talk here.
0:28:33.01 –> 0:28:35.47 I’ll do that right now, in case you want to check out
0:28:35.47 –> 0:28:38.283 some of the publications that I’m linking in this talk.
0:28:40.66 –> 0:28:42.33 Okay, so how do we do this estimation?
0:28:42.33 –> 0:28:44.27 And we use that penalized NCM algorithm,
0:28:44.27 –> 0:28:46.51 where in each step we’re drawing from the posterior
0:28:46.51 –> 0:28:47.99 with respect to the random effects, given
0:28:47.99 –> 0:29:50.07 the current aspects of the parameters,
0:29:00.07 –> 0:29:50.07 and the observed data, using Metropolis point of Gibbs.
0:28:55.18 –> 0:28:58.262 In the R packets, I’m going to talk about in a little
0:28:58.262 –> 0:29:03  bit, we update this to using a Hamiltonian Monte Carlo,
0:29:03 –> 0:29:03.98 but in the original version,
0:29:03.98 –> 0:29:06.27 we use Metropolis point of Gibbs, where we skipped
0:29:07.12 –> 0:29:09.36 components that had zero variance from the M-STEP.
0:29:09.36 –> 0:29:11.938 And then we use, in the M-step,
0:29:11.938 –> 0:29:13.94 two conditional maximization steps
0:29:13.94 –> 0:29:17.11 where we first update data, given the draws
0:29:17.11 –> 0:29:20.2 from the E-step, and the prior estimates for gamma
0:29:20.2 –> 0:29:23.74 and then up to gamma using a group penalty.
0:29:23.74 –> 0:29:25.4 So we use a couple of other tricks
0:29:25.4 –> 0:29:27.06 to speed up performance here.
0:29:27.06 –> 0:29:28.53 I won’t go too much into the details there,
0:29:28.53 –> 0:29:31.713 but you can check out the paper for more detail on
0:29:31.713 that.
0:29:33.33 –> 0:29:34.57 But with this approach, one of the things
0:29:34.57 –> 0:29:36.579 that we were able to show was that we have
0:29:36.579 –> 0:29:39.29 similar conclusions regarding bias and prediction er-
0:29:39.29 –> 0:29:41.42 as in the simple setup we had before,
where in this particular situation, we’re simulating a bunch of predictors that do not have any association with outcome, either 10 to 50 extra predictors, or there’s only two that are actually truly relevant. And so the prediction error in this model after this penalized selection process is generally the same, if not a little bit worse. And one thing that we find here is that the parameters are selected by the individual study approach we’re applying now at penalized distribution regression model has a low sensitivity to detect the true predictors, and a higher false positive rate in terms of selecting predictors that aren’t associated with outcome and simulation. And what we find here also is that the approach we developed had a much better sensitivity compared to other approaches for selecting the true predictors when accounting for study-level homogeneity, and the lower false positive rate as well. The example data sets that I talked about before, the four ones that I showed a figure up earlier, we did a whole data study analysis where we trained on three studies and held out one of the studies. We found that, you know, the approach that we put forward that put combining the data using our TSP approach, and then training a model using the pGLM had the lowest overall holdout study error compared to the approach using just
a regular generalized linear model, and then also the individual study approach as well. And we also compared it to another post called the Meta-Lasso, which we were able to adapt to do prediction, and we didn’t see that much improvement.

But in general, the result that we saw here was that the individual study approach had bad prediction error also across the different studies. So again, this sort of takes what we’ve already seen in the literature in terms of inconsistency, in terms of the number of genes that are being selected in each of these models, and also the variations in the prediction accuracy, this sort of reflects what we’ve been seeing in some of this prior work.

In order to you implement this approach in a more systematic way, my student and I, Hillary worked, put together an R package called The GLMMPen R Package. So this was just recently submitted to Journal of Statistical Software, but if you want to track the code, it’s available on Github right here, and we’re in the process of submitting this to CRAN as well.

This was sort of like a nice starter project that I gave to Hillary to, you know, get her feet wet with coding, and she’s done a really great job, you know, in terms of putting this together.

And some of the distinct differences between this and what we put forth in the paper is the use
of Hamiltonian Monte Carlo and the east app, instead of the Metropolis Gibbs. It’s much faster, much more efficient. We also have added helper functions for the (indistinct) tuning parameters, and also making some diagnostic plots as well, after convergence. And we’ve also implemented some speed and memory improvements as well, to help with usability.

Okay, so we talked about some issues regarding data integration, and then issues with normalization, how that impedes, or can impede validation in future patients, and then we introduced a way to sidestep the normalization problem, using this sort of rank-based transformation, and an approach to select consistent predictors in the presence of between-study heterogeneity. So next, I’m going to talk about a case study in pancreatic cancer, where we took a lot of these tools, and applied them to a problem that some collaboratives of mine were having, you know, at the cancer center at UNC. And to give a brief overview of pancreatic cancer, it has a really poor prognosis. Five-year survival is very low, you know, typically 5%. The median survival tends to be less than 11 months, and the main reason why this is the case is that early detection is very difficult, so when patients show up to the clinic, they’re oftentimes in later stages, or gone metastatic. So for those reasons, it’s really important to place
patients on optimal therapies upfront, and choosing the best therapies, specifically for a patient, you know, when after they’re diagnosed. So breast and colorectal cancers have long-established subtyping systems that are oftentimes used. Again, an example of a few of them in breast that have actually been approved by the FDA for clinical use, but there’s nothing available for, in terms of precision medicine for pancreatic cancer, except for a couple of targeted therapies for specific mutations. So in 2015, the Yeh Lab at UNC, using a combination of non-negative matrix factorization and consensus clustering, where it was able to discover two potentially clinically applicable subtypes in pancreatic cancer, which they call basal-like, the orange line here, which has a much worse survival compared to this classical subtype in blue, where patients seem to do a little bit better. And so with this approach, they used this unsupervised learning, set of learning techniques to derive these novel subtypes. And so when they took these subtypes and overlaid them from data from a clinical trial where they had treatment response information, they found that largely patients who with basal-like subtype tended to have tumors that did not respond to common first-line therapy, Folfirinox.
Their tumors tended to grow from baseline. Whereas patients that were the classical subtype tended to respond better on average compared to the basal samples. So the implications here are that if you are, subtype is basal, you should avoid Folfirinox at baseline entry with an alternative type drug, typically Gemcitabine and nab-paclitaxel Abraxane. And then for classical patients, they should receive Folfirinox. But the problem here is that subtyping clearly is an unsupervised learning approach, right? It’s not a prediction tool. So it’s, this approach is quite limited if it, when you have to do, assign a subtype in a small number of patients, it just doesn’t work. So what some people have done in the past, so they simply take new patients, and recluster them with existing, their existing training samples. The problem with that is that the subtype assignments for those original training samples might change. The goal here was to leverage the existing training data that’s available to the lab, which come from different platforms to come up with an approach, a classifier to predict subtype, given a new patient’s genomic data, to get subtype,
0:36:23.394 –> 0:36:24.89 a predicted subtype for that individual.
0:36:24.89 –> 0:36:28.41 So of course, in that scenario, we also want to make sure
0:36:28.41 –> 0:36:30.67 that that process is simplified, and that we make
0:36:30.67 –> 0:36:32.76 this prediction process as easy as possible,
0:36:32.76 –> 0:36:36.157 in the face of all these issues we talked about regarding
0:36:36.157 –> 0:36:39.78 normalization and the training data to each other,
0:36:39.78 –> 0:36:42.44 and also normalization of the new patient data
0:36:42.44 –> 0:36:43.94 to the existing training data.
0:36:45.26 –> 0:36:48.82 So using some of the techniques that we just talked about,
0:36:48.82 –> 0:36:50.76 we came up with a classifier that we call PurIST,
0:36:50.76 –> 0:36:53.43 which was published in the CCR last year,
0:36:53.43 –> 0:36:56.27 where essentially we were able to do that.
0:36:56.27 –> 0:36:59.17 We take in the genomic data for a previous patient,
0:36:59.17 –> 0:37:04.17 and able to predict subtype based off of that,
0:37:04.18 –> 0:37:05.8 the train model that we developed.
0:37:05.8 –> 0:37:08.754 And in this particular paper, we had nine data sets
0:37:08.754 –> 0:37:10.75 that we curated from the literature, three of which
0:37:10.75 –> 0:37:12.578 that we used for training,
0:37:12.578 –> 0:37:13.54 the rest we used for validation.
0:37:13.54 –> 0:37:16.4 And we did consensus clustering on all of them,
0:37:16.4 –> 0:37:18.11 using the gene list that was derived
0:37:18.11 –> 0:37:20.978 from the original publication,
0:37:20.978 –> 0:37:22.8 where the subtypes were discovered to get labels,
0:37:22.8 –> 0:37:25.18 subject labels for each one of the subjects
0:37:25.18 –> 0:37:26.82 in each one of these studies.
0:37:26.82 –> 0:37:30.37 So once we had those labels from consensus clustering,
0:37:30.37 –> 0:37:33.17 we then merged the data from our three largest studies,
0:37:33.17 –> 0:37:34.97 which are our training studies.
0:37:34.97 –> 0:37:37.34 We did some sample for filtering based on quality,
and we filtered some genes based off of, you know, expression levels and things like that.

And then we applied our previous training approach to get a small subset of top scoring pairs from the data.

And in this case, we have eight that we selected, each with their own study-level coefficient.

And then for prediction, the process is very simple, we just check in that patient, whether gene A is greater than gene D for each of these pairs, and that gives us their binary vector of ones and zeros.

We multiply that by the coefficients from the train model.

This is basically just calculating a linear predictor from this logistic regression model.

And then we can convert that to a predicted probability of being basal.

So using this approach, we were able to select 16 genes pertaining to eight subtypes, but we can find here that the predictions tend to coincide very strongly with the labels that were collected using consensus clusters.

So that gives us some confidence that reproducing in some way, you know, this, the result that we got using this clustering approach.

You can also clearly see here that as the subtype changes, you see flips in the expression in each one of the pairs of genes that we collected in this particular study.

And then when we applied this model
to six external validation dataset, we found that it had a very good performance in terms of recapitulating subtype, where we had a relatively good sensitivity and specificity in each case, which we owe part to the fact that we don’t have to worry as much about this sort of cross-study normalization training time. When we looked at the predictive values in these holdout studies, the predictive subtypes, we recapitulated the differences in survival that we observed in other studies as well, where basal-like patients do a lot worse compared to classical patients. If you want to look a little bit more at the details in this paper, you can check out this link here, and if you want to access the code that we used to make these predictions, that’s available on this Github page at this link right here. Another thing that we were able to show is that for patients that had samples that are collected through different modes of collection, whether it was bulk, FNA, FFPE, we found that the predictions in these patients tend to be highly consistent, and this is basically deriving itself, again, from the simple assumption behind TSPs, where the relative rank within the subject of the expression
0:40:13.06 –> 0:40:14.99 of these genes is predicted.
0:40:14.99 –> 0:40:17.31 So as long as that is being preserved,
0:40:17.31 –> 0:40:21.44 then you should be able to have the model predict well
0:40:21.44 –> 0:40:23.289 in different scenarios.
0:40:23.289 –> 0:40:27.63 So when we also went through CLIA validation for this tool,
0:40:27.63 –> 0:40:31.154 we also confirmed 95% agreement between replicated runs
0:40:31.154 –> 0:40:36.154 in other platforms, and we also confirmed concor-
dance
0:40:37.95 –> 0:40:42.77 between NanoString and RNAC, also through different modes
0:40:42.77 –> 0:40:43.603 of sample collection.
0:40:43.603 –> 0:40:46.69 So right now this is the first clinically applicable test
0:40:46.69 –> 0:40:50.61 for a prospect of first line treatment selection in PDAC.
0:40:50.61 –> 0:40:54.25 And right now we do have a study that just recently opened
0:40:54.25 –> 0:40:56.39 at the Medical College of Wisconsin that’s using PurIST
0:40:56.39 –> 0:40:58.39 for prospect of treatment selection,
0:40:58.39 –> 0:41:01.97 and we have another one opening at University of Rochester,
0:41:01.97 –> 0:41:06.32 and also at UNC soon as well.
0:41:06.32 –> 0:41:09.51 So this is just an example about how you can take
0:41:09.51 –> 0:41:14.04 a problem, you know, in, from the literature,
0:41:14.04 –> 0:41:17.57 from your collaborators, come up with a method,
0:41:17.57 –> 0:41:22.15 and some theory behind it, and really be able to come up
0:41:22.15 –> 0:41:24.31 with a good solution that is robust,
0:41:24.31 –> 0:41:27.44 and that can really help your collaborative
0:41:27.44 –> 0:41:29.763 at your institution and elsewhere.
0:41:31.85 –> 0:41:33.51 Okay, so that was the case study.
0:41:33.51 –> 0:41:34.56 To talk about some current work
that we’re doing just briefly.

So we wanted to think about how we can also scale up the,

desired framework that we developed for the pGLMM,

and one idea that we’re pursuing right now

with my student Hillary, is that we’re thinking about using, borrowing ideas from factor analysis
to decompose, do a deep, deterministic decomposition
of the random effects to a lower dimensional space,

where essentially, we can essentially map
between the lower dimensional space (indistinct) factors,

which is r-dimensional, to this higher dimensional space,

using some by matrix B, which is q by r,

and essentially in doing so, this reduces the dimension
of the integral in the Monte Carlo EM algorithm.

So rather than having to do approximate integral
and q dimensions, which can be difficult,
you can work in a much lower space in terms of integral,

and then have this additional problem
of trying to estimate this matrix,

and not back to the original dimension cube.

So that’s something that we’re just starting to work on
right now, and another thing that we’re starting to work on

is the idea of trying to extend some of the work
in variational autoencoders

that my student David is working on now.

His current work is trying to account for missing data
when trying to train these sort of deep learning models,

the VAEs unsupervised learning model’s oftentimes used for dimensional reduction.

You might’ve heard of it in single cells sequencing applications.

But the question that we wanted to address is, well, what if you have missing data, you know, in your input features X, which might be (indistinct)?

So essentially we were able to develop input.

So we have a pre-print up right now, it’s the code, and we’re looking to extend this, where essentially, rather than worrying about this latent space Z, which we’re assuming that that encodes a lot of the information in the original data, we replaced that with learning the posterior of the random effect, given the observed data.

And then in the second portion here, we replaced this generative model with the general model of y given X in the random effects.

So that’s another avenue that can allow us to hopefully account for non-linearity, and arbitrator action between features as well.

And also it might be an easier way to scale up some of the analysis we’ve done too, which I’ve already mentioned.

Okay, so in terms of some concluding thoughts, I talked a lot about how the original subtypes were derived for this pancreatic cancer case study using NMF.
and consensus clustering to get two subtypes. But there were also other groups that are published, subtyping systems, that in one, they found three subtypes, and in another one they found four subtypes.

So the question is, well, you know, well, which one do we use? Again, this is also confusing for practitioners about which approach might be more meaningful in the clinical setting. And each of these approaches were also derived using NMF and consensus clustering, and they were done separately on different patient cohorts at different institutions.

So you can see that this is another reflection of heterogeneity in single-study learning, and how we can get these different or discrepant results from applying the same technique to 200 genus datasets that were generated at different places. So of course this creates another problem, you know, who’s right, which approach do we use? And it’s kind of like a circular argument here. In the paper that I mentioned before with PurIST, another thing that we did is we overlaid the others subtype system calls with the observed clinical outcomes for the studies that we collected. And one of the things that we found was that, each of them also had something,
something that was very similar to the basal-like subtype, and for the remaining subtypes, they had survival that was similar to the classical subtype. So one of the arguments that we made was that, well, if the clinical outcomes are the same for the other subtypes, you know, are they exactly right necessary for clinical decision-making?

That was one argument that we put forth. And when we looked at the response data, again, we saw that one of the subtypes in the other approaches also overlapped the basal-like subtype in terms of response. And then for the remaining subtypes, they were just kind of randomly dispersed at the other end, of the spectrum here in terms of tumor present, tumor change after treatment. So the takeaway here is that heterogeneity between studies also impacts tasks in unsupervised learning, like the NMF+ consensus clustering approach to discover subtypes. And what this also does is, as you can imagine, injects a lot of confusion into the literature, and can also slow down the process of translating some of these approaches to the clinic. So this also underlies the need for replicable cross-study sub discovery approaches, for replicable approaches for unsupervised learning.
That’s something that, you know, something that we might, we hope to be working on in the future, and we hope to see more work on as well. So to summarize the, one of the major points of this talk was to introduce and discuss, you know, replicability issues in genomic prediction models, supervised learning, that stems from technical, and also non-technical sources. We also introduced a new approach to facilitate data integration and multistory learning in a way that captures between-study heterogeneity, and showed how this can be used for the prediction of subtype for pancreatic cancer, and also introduced some scalable methods and future direction in replicable subtype discovery.

I just want to thank some of my faculty crowd, Quefeng Li, Junier Oliva from UNC computer science, Jen Jen Yeah from surgical oncology at Lineberger, Joe Ibrahim as well, UNC biostatistics, and also my students, Hilary, who’s done a lot of work in this area, and also David Lim, who’s doing some of the deep learning work in our group. And that’s it, thank you.

Does anybody here have any questions for the professor? Or anybody on the, on Zoom, any questions you want to ask? It looks like I’m off the hook.
All right, well, thank you so much.

Really appreciated your talk.

Have a good afternoon.

All right, thank you for having me.