Project 1: What Makes Negotiations Successful and Unsuccessful?

1 The problem setting

This project is related to work by UMD social psychologist Prof. Michele Gelfand and her colleagues on the effectiveness of intercultural negotiations. What makes intercultural negotiations successful or unsuccessful is poorly understood, and there has been relatively little work directly on this problem. In a recent paper, Imai and Gelfand (2010) explored this question by conducting a set of simulations in which dyads (i.e. pairs) of people, one American and one East Asian, engaged in negotiations about a realistic but artifical scenario involving a joint business venture (opening a store that would sell both wine and groceries, where issues to be negotiated included things like what time the store would open, how much floor space each would have, and the average temperature of the store).[1]

For each dyad, Imai and Gelfand collected metadata about the participants, including, among others, measures of extraversion, openness, emotional intelligence, cultural intelligence, and how long the East Asian person had spent in the U.S. They then recorded and transcribed the two people negotiating with each other. And — with considerable effort — they *coded* the dialogues. Social scientists use the word *coding* to mean the same thing we NLP people call *annotation*. That is, they came up with relevant labels for parts of a negotiation conversation, like *OM (multi-issue offer)*, and assigned those labels to pieces of the dialogue.

For example, consider the following snippet of conversation:

1. Person 1 (Grocery): If I offered you 20,000 for renovation. . . If I was willing to do 8:30 but floor space you offered me 70%, and temperature, if you offered me 71 degrees, but grand opening date maybe be July 1st? [Coded as *OM (multi-issue offer)*]

2. Person 1 (Wine): Well I mean, for me though, renovation and temperature is sort of more important. . . I mean. . . [Coded as *IR (priority information)*].

There is a detailed coding scheme, involving 60 or so distinct labels. A single speaker turn might be broken up into several "thought units", each one with its own code. For example, here are two thought units from a single speaker turn, and each thought unit has its own code (IP and PC, respectively).

> Yeah. So I think the the later time is better for me. [IP]

[1] The PDF for their article is at http://www.bsos.umd.edu/psyc/gelfand/Imai&Gelfand2010.pdf and a fully detailed writeup can be found in Imai's thesis at http://drum.lib.umd.edu/handle/1903/6745.

> I'm not sure whether we can negotiate about this and reach a time and schedule that – that would be good for both of us. [PC]

Interestingly, there is an objective measure of success for these negotiations. From the article:

> This simulation involved a mixed-motive negotiation between a specialty grocery shop owner and a wine shop owner. Participants were told that a successful real estate company has proposed developing a multifunctional market that consists of a wine store and a grocery store under one roof with common décor but with separate areas for their respective top-quality products. Participants were told that they were seriously interested in the shared market but needed to negotiate five unresolved issues with the other vendor: (1) Hours of Operation, (2) Renovation Costs, (3) Floor Space, (4) Temperature, and (5) Grand Opening Date. Participants were given a payoff schedule that listed the possible levels of settlement on each of the five negotiation issues, and the number of points associated with each level of settlement that indicated the amount of worth of that level of settlement to the negotiator.

The success of a negotiation was measured by the joint profit achieved on the payoff schedule, as determined by how the negotiators resolved the five negotiation issues.

As it was structured, this was a classic kind of study involving a set of independent variables and their ability to predict a dependent variable (joint profit). Imai and Gelfand (2010) did a number of regression analyses to look at the effect of different characteristics of the negotiators and the negotiations themselves. Among other things, they established cultural intelligence (CQ, or "cultural quotient") as a factor that improves intercultural negotiation outcomes, and they showed, perhaps surprisingly, that a number of other individual factors like emotional intelligence and openness did not significantly improve the ability to negotiate successfully.

2 The goal

The Imai and Gelfand (2010) study relied on human coding, a very labor-intensive process, and it also was quite shallow in terms of what properties of the language were used in the analysis. We have a reasonable amount of data to work with — 75 coded negotiation dialogues, each one typically 100-200 "thought units" in length — and it seems as if it should be possible to use statistical NLP to do interesting things. The task, quite simply, is to apply ideas from class to do something interesting with these data, either to help the social scientists do their work, or to discover something interesting about the negotiations.

The Imai and Gelfand data should be treated as confidential. Please do not redistribute.

Some ideas:

Automatic coding of individual turns. Can we train automatic annotators for some or all of the relevant labels? One way to approach this would be as a text classification task: treat each thought unit as x and its associated label as y. Create a set of feature functions $f_i(x)$, train a supervised classification algorithm, and test on held out test data (or use cross-validation, since the quantity of available data is relatively limited). If you were to succeed at producing correct codes a reasonable percentage of the time, even for only a subset of the possible codes (perhaps the several most frequent?), we could change the human coder's task in the future to one of reviewing and correcting the classifier's output rather than coding from scratch.

Thinking in more detail about the real-world use case, it would be interesting to threshold based on classifier confidence, so that the system only offers a label in cases where it is confident; this would make it possible to analyze the tradeoff between accuracy and coverage. (See discussion in the Resnik and Lin evaluation chapter.) Also, it might make sense to evaluate "accuracy at n" for, say, n=1, 3, 5". In this kind of evaluation, the system's response is a k-best list, sorted from most to least confident, and the system gets credit when the correct answer is one of the top n responses. ("Accuracy at 1" is simply the accuracy.) Accuracy at n is useful because it helps approximate the extent to which the system would help a human coder by narrowing the number of possible responses down to only n to choose from.

Automatic coding of entire negotiations. Coding each turn in isolation throws away some valuable information. Like part-of-speech tagging, coding these negotiations is a *sequence labeling* task, where information about the surrounding labels might contain useful information. Unlike part-of-speech tagging, though, if you think about this as a generative process, the hidden tag generates not a single word, but an entire thought unit, which contains a *set* (really a *bag*) of words. This means HMMs are likely to be a poor approach to take. However, there are a variety of other algorithms for sequence labeling tasks, of which perhaps the most obvious to try would be Conditional Random Fields (CRFs). We'll be covering these in class; also see Hannah Wallach's nice Web page on CRFs, with a whole variety of useful references, at http://www.inference.phy.cam.ac.uk/hmw26/crf/.

Improving prediction using feature engineering. There's another interesting supervised task embedded in the Imai and Gelfand (2010) work, where the x is the entire negotiation dialogue, and y characterizes *how successful* the dialogue was. FYI, this is the one that the social scientists are most excited about, though the automatic coding stuff above, is certainly of interest, also.

Recall that y in their study was a continuous value, the joint profit attained by the pair of people negotiating. This means that one natural approach (the one taken by Imai and Gelfand) is to treat the problem as a *regression* task. One could also turn it into a classification problem by thresholding, e.g. any dialogue where the joint profit is 1300 or below gets labeled y =LOW, if joint profit is 1300 to 1399 then y =MEDIUM, and if joint profit

is 1400 or higher then y =HIGH. (These thresholds split this set of dialogues into roughly even thirds.) As a first pass, you could even throw out the MEDIUM category, just to try to discover features that distinguish the more extreme cases. Then, time permitting, you could add the medium cases back in and see what happens, and/or use the features that were useful to attack the regression problem where y is the joint profit.

Either way, whether it's a regression problem or a classification problem, it would be interesting to (a) explore what happens when you use the features used by Imai and Gelfand in their article, and (b) try adding new features of your own. You could, for example, use unigrams or bigrams as features. You could use association measures to discover strongly associated "bigrams" at the level of the human codes, and see whether those might be useful features. You could include word-based features (unigrams, n-grams). You could explore whether differential use of pronouns (e.g. *I* and *you* versus *we*) has predictive value. Or use of words with positive or negative sentiment. Or use of words in a whole variety of other categories.[2]

Sequential pattern mining. As noted earlier, these negotiations can be viewed as labeled sequences. This raises the interesting possibility that there might be interesting sequences of labels associated with successful and unsuccessful negotiations. For example, perhaps if you see a proposal, followed by a hostile response, followed by a counter-proposal, followed by a request for information, this is more likely to be a negotiation that will succeed. (I made that example up, but you get the idea.) CRFs are one way to explore this, as above. Although we're not looking at it in this class, apparently there is a literature on *sequential pattern mining* (see also *sequential data mining*), with applications in, e.g., analysis of search engine query logs, financial stream data, and pattern discovery in genomic DNA sequences in bioinformatics. Discovering such patterns could be quite interesting, particularly if they turned out to be useful as predictive features in regression or classification as discussed above. I don't have as much to say about this approach, never having explored it myself, but it might be worth a look.

3 Resources

We will provide you with the negotiation dialogues and associated metadata.

We will provide a variety of lexical resources mentioned above, e.g. Pennebaker's LIWC dictionary, the MPQA sentiment lexicon, etc. I strongly encourage sharing of discovered resources across groups.

[2]Psychologist Jamie Pennebaker has gotten a lot of mileage out of differences in pronoun use, e.g. see http://www.nytimes.com/2008/10/14/science/14prof.html. We have a version of the lexicon from his Linguistic Inquiry and Word Count (LIWC) program that you could use. (It's pronounced "Luke".) LIWC also contains sentiment-word categories (positive and negative emotion words). We also have an independently constructed sentiment-word lexicon, the MPQA lexicon from University of Pittsburgh.

You can find a detailed description of the coding system in Imai's thesis. There is also a detailed coding manual that we can provide, which is what the human coders used.

All the dyads are cross-cultural and contain one person who is American, and one person who is East Asian. In case it might be useful to know which is which, we will provide a file that identifies this information for the first speaker in the negotiation, in this format:

```
<Dyad No 2> American
<Dyad No 3> East Asian
<Dyad No 4> American
```

There are a variety of good machine learning resources you can use, and I strongly recommend *against* coding up machine learning algorithms from scratch if you can avoid it. For example:

WEKA. The Weka toolkit (http://www.cs.waikato.ac.nz/ml/weka/) is one of the best known and most widely available machine learning packages. It supports a wide range of supervised learning techniques, including most of the ones we have discussed in class. Weka comes with both a graphical user interface and a command-line interface, as well as a java API. The basic idea, in using Weka, is to represent your learning problem using an "ARFF" file, within which each instance is represented as a feature vector. (The header of the file identifies the types of the features as well as the feature that constitutes the class being predicted.) Once you've got your data into the .arff format, it's very easy to try out different learning algorithms and/or different parameters for the same algorithm.

MALLET. The MALLET toolkit (http://mallet.cs.umass.edu/) is another very nice package. It's not as well documented as Weka, but it has some decent "quick start" Web pages with useful examples, fairly readable java source code (or so I'm told), and a very active online discussion group monitored by MALLET team members who actually seem to be pretty responsive. MALLET overlaps with Weka a bit, but it also supports sequence modeling (conditional random fields, a.k.a. CRFs) and unsupervised topic modeling (Latent Dirichlet Allocation, a.k.a. LDA). And again, you can get your data into a fairly standard format and play with different parameterizations for learning, there's a java API, etc. (But no GUI.)

Others. There are a variety of other toolkits out there for specific approaches such as maximum entropy modeling, support vector machines, and decision tree learning, e.g. see LingPipe (http://alias-i.com/lingpipe/). It also appears that NLTK (http://www.nltk.org/) offers useful machine learning toolkit functionality (decision tree, maximum entropy, naive Bayes clasifiers, and an interface to Weka) although I'm not particularly familiar with it. There are various other lists of machine learning toolkits out there; probably one of the

best too look at would be Hal Daume's list of useful machine learning links and software at `http://www.cs.utah.edu/~hal/courses/2008F_ML/`.

Finally, let me observe that there are some interesting *Bayesian modeling* approaches out there, which might be interesting to try if you're particularly ambitious and want to go beyond existing off the shelf classifiers and models. We'll cover basics of Bayesian modeling in class, albeit a little later in the semester. There are a number of toolkits that support Bayesian modeling; ask me for info if you're interested.

4 What you need to do

Project plan. On a date that I specify (probably a week or two after this assignment is handed out), I'm going to want to see a proposed project plan. This should describe what your group plans to try in enough detail that I can provide you feedback and guidance, and particularly so that I can steer you away from approaches that are likely to get you bogged own. Following the outline of the project writeup would be a good way to organize this. Note that you don't need absolutely every detail worked out; just give me enough so I can see what you're aiming for. And if the writeup format isn't a good match for what you're going to do, that's ok, too, it's fine to diverge from it. What's important is that you're getting an early start on thinking through the issues and your plan and giving me a chance to comment.

Project writeup. Here's what I expect to be turned in by each group on the due date.

1. A tarball/zipfile of your source code, *including enough information for someone to run it without having to ask you questions.* This should include a README that walks the reader through what to type on the command line to train, evaluate, etc. For example:

```
This code builds a regression model to predict joint profit.
See our writeup for details of the features we use, the
regression model, etc.  We have successfully run this code
on Linux, but haven't tried it on a Mac or Windows.

Dependencies:
  We ran this using perl version 5.8.8
  Make sure MALLET is installed on your machine and in your path.
  CPAN perl modules you need to have installed:
     Text::CSV - handles CSV files
     Text::English - has the Porter stemmer

To train:
```

Run this from our top-level directory (parent of resources/):

```
% train.pl --corpus=negotiation_corpus/merged_turns.csv
           --metadata=negotiation_corpus/metadata.csv
           --exclude-dyads=74,75
           --use-features=unigrams,pronouns,liwc358,cq,iq
           --run-ID=test123
```

This will train a model as described in our writeup, using every dyad except 74 and 75, and it will create a new directory test123 containing the model files and other files we create along the way. (It'll barf that directory already exists.) Takes a minute or two on a Mac. Status messages during training (e.g. "Importing data into MALLET", "Starting MALLET training") will appear on STDERR.

To test on unseen data:

```
% test.pl --corpus=negotiation_corpus/merged_turns.csv
          --metadata=negotiation_corpus/metadata.csv
          --test-dyads=74,75
          --run-ID=test123
          --outfile=test123_output.csv
```

This will run the model in directory test123 on dyads 74 and 75. It will produce an outfile, test123_output.csv, that contains one line per test dyad, showing

```
Dyad-number  True-joint-profit  Predicted-joint-profit  MSE
```

where MSE is the mean squared error, i.e. (true-predicted)^2. At the end of the file there is a separate table giving a summary score for this test set, which is just the average and standard deviation of the MSE taken over all the test dyads, along with the specs for this experiment, e.g.

```
SUMMARY   test123_output.csv    74,75         172.4   14.6
```

That way you can do a bunch of experiments, and just grep SUMMARY from all the output.csv files to get a comparison of how we did on different feature sets. Including the standard deviation makes it easy to do statistical significance tests comparing different results.

To run a cross-validation experiment:

```
% crossvalidate.pl
          --corpus=negotiation_corpus/merged_turns.csv
```

```
--metadata=negotiation_corpus/metadata.csv
--use-features=unigrams,pronouns,liwc358,cq,iq
--folds=5
--seed=13
--run-ID=test234
--outfile=test234_output_crossval5.csv

This will run a 5-fold cross-validation experiment, breaking the
dyads into 5 folds randomly, seeding the random number generator
with 13 so that runs are 100% comparable and reproducible. Messages
on STDERR will show you how the run is progressing.  The crossval
output file gives the summary information for each individual
fold plus the mean and standard deviation of the MSE.
```

Don't feel like you need to mimic everything here. I happen to code in perl, and I like command line arguments. But you might just as well use Java, python, or Makefiles to drive your experimentation, you might put all the options into a separate configuration file for each run rather than on the command line, etc. Do what makes sense, just look at the example above as guidance about the kind of information I'd like to see, and what I mean by a README that lets me run your code.

2. A writeup including the elements below. Please stick with this main structure, though you can add sections if you need to. (Also, I recommend you look at some NLP conference papers to get a general feeling for how research descriptions tend to be structured and written.)

   The group grade will be based primarily on the writeup. So make sure you produce something clean and well written and don't leave it until the last minute. Also, a common problem I've seen is breaking the writeup into sections, giving each person a section, and then simply throwing them together at the end. This results in really uneven quality and writeups that are sometimes quite hard to read. There's a difference between a group effort and a union (or concatenation!) of individual efforts. Aim for the former, not the latter.

   (a) Who is in this group. Optionally, a rough breakdown of people's roles in the project, if things were broken out that way. E.g. one person might be implementing a set of feature extraction algorithms, another might have focused on experiment-running infrastructure, another on different classification or regression algorithms, etc. If things weren't broken out that cleanly, that's ok, too, I'm mainly curious about how you organized things.

   (b) Preprocessing details and other information on what you did to/with the data in order to work with it. (Lowercasing, tokenization, stemming, lemmatization, etc.)

   (c) What method or methods you used for classification/regression/modeling. You do not need to regurgitate descriptions of the algorithms, though it would be

helpful if you commented on why you made the particular choices you made. I'm ok with you using just one kind of algorithm, though for many of these packages it's just as easy to try two or three and if you do so, it gives you the opportunity to see a general pattern of results across classifiers or models.

(d) What features you used, and the rationale behind them. This is a real opportunity for thought, creativity, exploration, etc.

(e) What you used as an evaluation metric or metrics.

(f) A description of what you used as a baseline or baselines. Note that for this project it's going to be difficult to produce upper bounds, so don't worry about that.

(g) How evaluation was done, e.g. cross-validation details.

(h) A table or tables of results, showing performance for the baseline(s), and for one or more variations. Charts and graphs are nice, too, but (a) focus on readability and understandability, not flashiness, and (b) please make sure everything will be understandable even if it's printed in grayscale rather than color.

(i) A section (or sections) discussing the results, e.g. error analysis, etc. This is a main place where I'm looking for thought, effort, insight. What worked and didn't work? What did you discover? What might or might not be useful to the social scientists coming out of this?

An important part of the evaluation is going beyond the numbers to an analysis of why things turned out the way they did. (A good analysis can be as important as a positive result, in terms of good research.) One form of analysis might be an error analysis for an individual classifier/featureset combination, trying to identify generalizations about what it does well or what it does poorly. Another typical form of analysis might break errors into false positives and false negatives, or into other buckets, in order to seek insight into what's working, what's not, and how it could be improved.

(j) A section discussing any particular difficulties or hurdles you encountered. Please feel free to include ways in which this assignment could be made better. (The fact that we're working with small datasets is a response to comments in previous years, where large-data problems made it difficult for students to try out everything they wanted to try.)

3. Separately, by e-mail, each person should send me and the TAs three ratings for their team members as described under "Grading", below. Please put "Compling2 ratings - YOUR NAME" in the subject line so these messages are easy to spot. Attach a file named `YOUR_NAME.ratings.csv` containing the following:

```
your_name    teammate1_name    rating1    rating2   rating3
your_name    teammate2_name    rating1    rating2   rating3
```

Yes, I want your name repeated, identically, in the first column. Please agree within your team on a consistent way to write everyone's name. This way we can concatenate all the files we receive and sort by the 2nd column to easily aggregate the ratings for each person.

5 Grading

The group will receive a grade-in-common out of 75 points. By now I think you have a decent sense of my criteria, and I've been pretty explicit above. Thoughtfulness, effort, looking at data and results to achieve insight— all these things carry a great deal of weight with me.

For the remaining 25 points, each team member should anonymously rate each other team member as follows – *making sure to look at the definitions below to see what the numeric scales are supposed to mean and how to calibrate them.* I'm serious about the calibration issue. If everyone simply gives everyone the top score on every criterion, I will probably kick this back to you and make you do it again, or at least ask you to write a description backing up why that was the outcome.

- Rating 1: Collaboration. Scale from 1 to 10 where 10 is highest.
- Rating 2: Contribution to success of the project. Scale from 1 to 10 where 10 is highest.
- Rating 3: Effort. Scale from 1 to 5 where 5 is highest.

Collaboration. 10 means that this person was great to collaborate with and you'd be eager to collaborate with them again, and 1 means you definitely would avoid collaborating with this person again. Give 5 as an *average* rating for someone who was fine as a collaborator, but for whom you wouldn't feel strongly about either seeking them out of avoiding them as a collaborator in the future.

Contribution. 10 means that this person did their part and more, over and above the call of duty. 1 means that this person *really* did not contribute what they were supposed to. Give 5 as an average, *they did what they were expected to* rating. Note that this is a subjective rating relative to what a person was *expected* to contribute. If five people were contributors and each did a fantastic job on their pieces, then all five could conceivably get a rating of 10; you would not slice up the pie 5 ways and give each person a 2 because they each did 20% of the work! It is your job as a group to work out what the expected contributions will be, to make sure everyone is comfortable with the relative sizes of the contributions, and to recalibrate expectations if you discover you need to. Try to keep

things as equitable as possible, but if one person's skills mean they could do a total of 10% of the work compared to another person's 15%, and everyone is ok with this, then both contributors can certainly get a score of higher than 5 if they both do their parts and do them well. If you need help breaking up tasks, agreeing on expectations, etc., I would be happy to meet with the group to assist in working these things out.

Effort. A rating of 3 should be average, with 5 as *superior effort* (whether or not they succeeded) and 1 as *didn't put in the effort.* A rating below 3 would not be expected if the person's contribution was 5 or better. If a person just didn't manage to contribute what they were expected to, but you think they did everything in their power to make it happen, you could give them a top rating for effort (the classic "got an A for effort") even while giving them a low contribution score.

6 A Final Note

This project is ambitious. It attempts to give you an experience doing something real, not just a textbook exercise. It's the first time we're trying it. That means that there might be unanticipated problems, situations where people do not receive inputs they need to get their part done, intra-team politics, interpersonal issues, and who knows what else – just like in the real world. It also means that something new and interesting might come out of it, which is pretty cool.

Unlike the real world, which is not very forgiving, this is a controlled setting that involves the guidance of an instructor, who *can* be very forgiving. Remember that the activity is, first and foremost, a collaborative learning activity, with the emphasis on *learning.* If there are problems or issues of any kind, let me know sooner rather than later, and I will help to get them worked out. Also feel free to use the mailing list or discussion forum: the presence of multiple teams does *not* mean that you are competing with each other. (I considered adding extra credit for the team with the best results, but I specifically decided against it because I would much rather see a spirit of collaboration not only within teams but at the level of the entire class.)

And remember to have fun!