Twenty Years of the Kansas Event Data System Project

Philip A. Schrodt
Dept. of Political Science
University of Kansas
Blake Hall
Lawrence, KS 66045
785.864.9024 (phone) - 785.864.5700 (fax)
Email: schrodt@ku.edu

VERSION 1.1: November 9, 2006
A copy of this document is available at
`http://web.ku.edu/keds/KEDS.history.html`

1 In the beginning there was event data

The importance of interactions in political behavior has been recognized since the beginnings of the efforts to systematically study international political behavior using "scientific" approaches. This research has primarily taken the form of the analysis of event data—nominal or ordinal codes recording the interactions among international actors as reported in the open presswhich break down complex political activities into a sequence of basic building blocks (e.g., comments, visits, rewards, protests, demands, threats, and military engagements). Event data sets were a major focus of some of the earliest quantitative international relations research in the 1960s and 1970s—work that generated Edward Azar's (1982) Conflict and Peace Data Bank (COPDAB) and Charles McClelland's (1976) World Event Interaction Survey (WEIS). Over the past decade, interest in event data analysis has increased as the combination of machine-readable news reports and automated coding have dramatically reduced the costs of generating, customizing, and analyzing event data.

Historically, event data have usually been coded by legions of bored undergraduates flipping through copies of *The New York Times*. Automated coding provides two advantages over these traditional methods:

- Coding can be done more *quickly* by machine than by humans; in particular the coding of a large data set by a single researcher in real-time is feasible;
- Machine coding rules are applied with complete *consistency* and are not subject to inter-coder disparities caused by fatigue, differing interpretations of the coding rules or biases concerning the texts being coded;

The disadvantages of automated coding are that it cannot deal with sentences having a complex structure and it deals with sentences in isolation rather than in context.

This article will discuss the development of the automated coding systems of the Kansas Event Data System (KEDS) project. I first discuss the gradual experimentation that produced, over a period of ten years, the first automated coding system—the eponymous KEDS computer program—that could produce data acceptable in refereed articles. This will be followed by discussions of subsequent work—KEDS's open-source successor TABARI and the CAMEO event coding scheme—as well as some comments on project management and funding, and finally some reflections on why this approach is not more widely used in academic research. This is a personal history rather than

an attempt to provide a fully balanced account, and is told from my perspective with an emphasis on the developments that seemed most important to me; other recollections may differ.

2 Roots

The enthusiasm for event data among researchers was tempered by the fact that it was very expensive to produce. Event data collection slowed in the late 1970s as funding—which had largely come from the U.S. Department of Defense Advanced Research Projects Agency—was discontinued. The existing data sets continued to be widely used—one study found them to be the third most frequently used form of data in quantitative international relations (IR) research—but they were not updated.[1]

I had done some limited work with event data in graduate school, and in the early 1980's Alex Mintz and I experimented with developing some event data forecasting methods using conventional conditional probability theory. This was presented at International Studies Association (ISA) meetings in 1984, and eventually published as Schrodt and Mintz (1988). Our key finding was that interactions involving Kuwait disproportionately triggered other responses. This finding received a mixed response at the ISA: Following the presentation, a U.S. political analyst came up and said "That result on Kuwait is the stupidest thing I've ever heard, and all you've done is show that event data are useless." He was followed shortly by an Israeli analyst who said "That result on Kuwait is completely obvious, and all you've done is show that event data are useless." Events involving Kuwait in 1991 and 1992 provided some vindication for our finding, though the "if you are right, it's obvious; if you are wrong, it's wrong" attitude continues to plague systematic modeling efforts in the policy community.

In the mid-1980s, I became involved in a fascinating set of work influenced by the then very trendy research on "artificial intelligence" (AI). A number of political scientists, dissatisfied with the limitations of conventional statistical analysis, attempted to apply AI to the formal study of international relations and foreign policy, an effort we labelled "AI/IR." This produced a variety of quite novel approaches using computational methods—before being tragically destroyed after exposure to a particularly virulent strain of post-modern deconstructionism, for which at the time there was very little acquired immunity.[2] My own efforts at computational modeling largely revolved around applications of event data, in particular trying to use these to simulate the problem of reasoning by analogy. Various efforts along these lines can be found in `http://www.ku.edu/~keds/papers.dir/Schrodt.PRL.2.0.pdf`.

I used an assortment of event data sets in this work, mostly versions of COPDAB and WEIS that I had acquired from various sources. It was clear, however, that these were not very dense, particularly on the Arab-Israeli conflict, the area where I was also doing field research. More critically, they were not being updated, so I could not use them to study contemporary issues or do true forecasting. I also did some work with Russell Leng's much more extensive (and to this day, still very under-utilized) BCOW data set. Leng's data, while human coded, actually provided a template for how the KEDS project coding would later develop, with its emphasis on specific conflicts (rather than trying to code the entire world), and an event coding framework that provided more detail than WEIS or COPDAB.[3]

[1] WEIS was primarily funded by DARPA and become the de facto coding scheme for Dept. of Defense coding work. The source of funding for COPDAB, on the other hand, was never clear, and the early COPDAB work was if anything more labor-intensive than WEIS. Edward Azar—who was of Lebanese origin— alluded to mysterious sponsors in the Middle East but was never specific. In retrospect, the most likely interpretation of this would be that the sponsorship was by the C.I.A., which has sponsored far more quantitative work than most people imagine, but almost never claims responsbility for this work. Azar died in 1984, an early victim of the AIDS epidemic, though COPDAB was maintained for another half decade or so by his graduate students.

[2] Sylvan and Chan (1984) provides early examples of this approach; Hudson (1991) has later, more sophisticated examples, and Trappl (2006) shows the approach is not dead yet.

[3] BCOW codes historical crises: my favorite event code in the system is "assume foreign kingship." BCOW also has a complex hierarchical system for describing the details of an interaction, where the fields vary depending on the type of event. BCOW allows, for example, tertiary actor coding in situations where the dyadic event is directed at a

In the process of doing some contract programming for a West Coast consulting firm developing a political decision-support system "for a major U.S. ally", I became acquainted with (and obtained a copy of) a WEIS data set that had been collected by various defense consulting firms for no less a client than the National Security Council in the Reagan White House, where event data was championed by a McClelland student named Richard Beale. This effort continued until Beale's untimely death in 1985.[4]

One of the novel features of this version of WEIS was the inclusion of brief English-language summaries of the news story that generated the event. This provided an opportunity to check whether it was possible, in principle, to go from a natural language text to event codes. Working with a Northwestern University undergraduate, David Leibsohn, we developed a simple computer program that mapped keywords to event codes, and presented this at the 1985 International Studies Association meetings (Schrodt and Leibsohn 1985). This produced credible results, and in particular provided early evidence that while some WEIS categories were subtle and difficult to code consistently (probably for humans as well as machines), many of the most common events—notably meetings and uses of force—were straightforward because they were described using a very distinct and specific vocabulary.

This system was *only* a proof-of-concept because it depended on the availability of short summaries. My intention at the time had been to eventually adapt this to code some sort of electronically-readable index of news stories, since an index would also involve very brief, literal texts. Large electronic databases were just beginning to become available, both by dial-up access (the Web was still a dream at this point) or via early CD-ROMs (which were horrendously expensive: CD-RW was also still a dream). However, the promises of these indices proved substantially more elaborate than their reality, and my efforts stalled for a couple of years.

In the late 1980's, the National Science Foundation undertook a major initiative titled "Data Development in International Relations" to update the most widely used international relations data sets (and, one suspects, reduce the grousing from the quantitative IR community over the resources going to the American National Election Survey). The first phase involved the Correlates of War data sets; the second event data.[5] This second phase was headed by my dissertation adviser Dina Zinnes and her colleague, the late Richard Merritt (Merritt, Muncaster and Zinnes 1994). A group of about twenty researchers was convened, and eventually NSF invested about $350,000 in a number of different event data projects.

Due to the advances in natural language processing that had occurring during the artificial intelligence boom of the 1980s, automated coding was now seen as a credible possibility, though by no means a certainty. DDIR therefore proceeded along two tracks—experiments with automated coding at Kansas and MIT, and a conventional human coding system, the Global Event Data System, that was based at the University of Maryland under Ted Robert Gurr and John Davies. GEDS advanced the human coding by providing a very elaborate coding framework—closer to BCOW than WEIS, although nominally based on COPDAB—and the by use of various coder-friendly tools on personal computers which, it was hoped, would substantially increase the speed of the coding.

My initial contribution on the automated coding front was a program named WINR—WINR is Not Relatus—that was an elaboration of the Schrodt-Leibsohn work but which now employed machine-learning methods. It worked reasonably, though not spectacularly, well, and its ultimate contribution was to simply provide some of the basic code for what would develop into KEDS, and demonstrate the limits of machine learning for this type of problem.

The name WINR was an inside joke playing against a natural language system being developed

specific third actor. Unfortunately this complexity, while appropriate to the description of interactions, has probably inhibited its wider use.

[4]Following the end of NSC sponsorship, responsibility for WEIS moved to McClelland's student Rodney Tomlinson at the U.S. Naval Academy, where the data set was maintained using student coders until about 1992. A composite of all of these efforts—colloquially known as the "WEIS grey data set"—has circulated informally in the quantitative IR community up until the present.

[5]A third phase would have focused on international political economy data but never got off the ground, in part because considerable data were already being made available by economic IGOs such as the World Bank and IMF

at MIT called RELATUS[6] and Richard Stallman's recursive "GNU is Not Unix." The machine-learning aspect was consistent with most of my work in AI/IR, but turned out to be a dead-end: KEDS and TABARI both eventually required extensive, and highly expert, dictionary development by humans. In retrospect, this simply reflected a general lesson from automated natural language processing in the 1980s—humans are so good at language, and language is such an idiosyncratic human construct, that it is better to let humans tell a machine what to do (and then have the machine routinely do it) than to try to develop machine learning algorithms.[7]

By the late 1980s, we had a reasonable set of techniques that provides credible results, but we had only shown that we could map natural language news *summaries* into event categories. This was a long way from the "holy grail" of producing data directly from news wire accounts. The closest thing available seemed to be various machine-readable indices, but these typically had insufficient information to resolve events beyond the level of the dozen or so major "cue categories" in the event coding schemes.

At this point I had a chance encounter with a student involved with University of Kansas (KU)[8] debate program who, on hearing about my research, asked "Why don't you just use Reuters?—it's available on NEXIS." "Where do I get NEXIS?" "At the Law School—all the debaters use it there."[9]

I arranged a meeting with one of the Law School librarians, who demonstrated that one could, in fact, download Reuters stories via a dial-up connection. He assured us that there was no marginal cost to the Law School for using the service, and then noted that the Law library was closed overnight. He then suggested we wait while he showed us a couple more things about the system, but he had to check with NEXIS technical support first. He placed a couple of calls, asking questions at the level of "Which one is the *any* key??" but in the process of getting authorization, repeated, loudly and slowly "Our NEXIS password is...".

Dial-up connection, library is closed, here's the password: we can take a hint. We set up a simple script to automate a log-in to NEXIS from about 2 a.m. to 5 a.m., and over the next several months downloaded tens of thousands of stories.

[6]RELATUS had a curious history. Sponsored by Hayward Alker, and taking advantage of the phenomenal social capital of the Artificial Intelligence Laboratory at MIT, it was probably the most sophisticated effort to come out of the AI/IR enterprise. However, it also had all of the weaknesses that eventually led to the "AI Winter" that followed the enthusiasm for AI in the mid-1980s. The system ran on specialized hardware—"LISP machines"—and while great promises were made for its potential—in particular, we were repeatedly reminded, it was going to work vastly better than anything that we lowly scum outside of MIT might even contemplate writing—it never got beyond the stage of a working prototype.

RELATUS quietly disappeared sometime in the early 1990s. The story told to me, albeit without a lot of detail, was that the system involved an incredibly complex data base that incorporated a lot of "self-modifying code," a typical characteristic of LISP systems, where the program code was treated as just another form of data. A programming error apparently caused the system to completely destroy itself, and for reasons that aren't clear, it could not be restored from backups. [A programmer failing to make regular back-ups?—nah...] The Cambridge-based "LISP machine" industry was collapsing at the same time because Moore's Law quickly gave $5000 off-the-shelf personal computers the same power as dedicated $50,000 LISP workstations, and RELATUS was never rebuilt in a portable form.

[7]The contemporaneous DARPA sponsored "Message Understanding Conference" project (ARPA 1993) had somewhat similar objectives—extracting details of terrorist incidents from news wire stories (and involving real computer scientists and real linguists!)—and came to similar conclusions: systems using extensive phrase dictionaries developed by humans far out-performed machine-learning systems.

[8]Why is the official abbreviation for the University of Kansas "KU" rather than "UK"?—as I've noted, human language is very idiosyncratic...

[9]Yes, kiddies, in those days NEXIS was a highly restricted resource, not something available on a browser at most research universities. We also walked to and from school every day in the snow. Barefoot. In June. Uphill. Both directions.

3 KEDS

Based on the WINR demonstration, DDIR provided a small $40,000 grant in 1991-1993 for the development of what became the KEDS program.[10] While the KEDS work for the DDIR project included some experimentation with German-language sources and foreign policy chronologies (Gerner et al 1994), most of our development focused on WEIS coding of interactions in the Middle East reported by the Reuters news service in English. We focused on this area both because it is very thoroughly covered in the international press, but also because we were doing field work in the area and could therefore cross-check the validity of event data based on our experience in the region.

KEDS relies on *shallow parsing* of sentences—primarily identifying proper nouns (which may be compound), verbs and direct objects within a verb phrase—rather than using full syntactical analysis. As a consequence it makes errors on complex sentences or sentences using unusual grammatical constructions, but has proven to be quite robust in correctly coding the types of English sentences typically found in the lead sentences of newswire reports. On early-1990s hardware, the system coded about 70 events per second, which seemed at the time to be a huge improvement over human coding projects, which typically have a sustained output of five to ten events per coder per hour.

KEDS had its professional debut at the 1992 International Studies Association meetings in Atlanta. The paper was being written, typically, at almost the last minute, and focused on a 12-year time series for the Arab-Israeli conflict. A number of different pieces had to come together—downloading and reformatting the Reuters stories, on-going dictionary development, and aggregation of the resulting events into an interval-level time series using the Goldstein (1992) scale—so only when the paper was nearly finished that could I actually look at the results. I still vividly remember finally getting the Israel-Palestinian series, and plugging it into MS-EXCEL to get a basic plot. My great fear was that the Palestinian intifada would not show up in the data. To my tremendous relief, there it was as a lovely (if noisy) spike followed by an exponential decay, the most conspicuous feature of the series.

One of the people who heard that ISA presentation was Doug Bond from the Program on Nonviolent Sanctions in Conflict and Defense at the Center for International Affairs at Harvard (Bond, Bennett, and Vogele 1994) who was beginning the development of a new event coding scheme, the Protocol for the Assessment of Nonviolent Direct Action (PANDA). The PANDA project worked in close collaboration with us for the next two years during the most intense development of KEDS in dictionary development, identification of bugs, and validation. The PANDA system coded a superset of the WEIS categories (160 categories versus the 63 categories in WEIS) that provided far more detail on nonviolent events, substate actors and internal interactions such as strikes and protests and also coded several contextual variables in addition to the standard date-source-event-target variables of event data.

The PANDA work eventually spun off a commercial event-coding operation—VRA, Inc. [`http://vra.com`]—which developed a coding program that used quite different principles than KEDS . The PANDA coding system, with the added collaboration of Craig Jenkins (Ohio State) and Charles Taylor (VPI) morphed into the Integrated Data for Events Analysis (IDEA) coding scheme (Bond et al 1997). Reuters reports dealing with the entire world have been coded for by VRA for 1985-2004; the resulting data set contains about 10-million events and can be downloaded from `http://gking.harvard.edu/data.shtml`

The accuracy of automated coding depends heavily on the source text, the event coding scheme and the type of event being coded. We have done a variety of different reliability checks using KEDS in early papers and articles. With Reuters lead sentences and the WEIS coding scheme, KEDS assigned the same code as a single human coder in about 75% to 85% of the cases. Approximately 10% of the Reuters leads have a syntactic structure that is too complicated or too idiosyncratic for KEDS to handle properly, although some of the residual coding disagreement comes from ambiguities

[10]The bulk of the DDIR funding went to the human coding of the GEDS project, which found, unfortunately, that the computer-*assisted* approach to event coding did not produce dramatic increases in efficiency. To the best of my knowledge, the data from the project was never used in any refereed publications.

in the WEIS coding categories themselves. KEDS had a somewhat lower agreement when compared to multiple human coders. In many cases this was not because KEDS was coding poorly but because KEDS was more consistent and less likely to miss multiple events in a single story than some of the human coders—in other words, the program was actually doing better than the humans (Schrodt and Gerner 1994). King and Lowe (2004), independently working with the VRA data, reached very similar conclusions, as have several unpublished studies. On the other hand, some people are skeptical whether this is even necessary: one referee reviewing our request to NSF for continued funding of KEDS wrote "In my experience the human coding of events is so bad I'm not sure you *want* to duplicate it."

In an experiment where dictionaries were optimized for the coding of a single day of Reuters leads, the PANDA project achieved a 91.7% machine coding accuracy; this probably represents the upper limit of accuracy for Reuters leads and a program using KEDS's shallow parsing approach (Bond, Bennett & Vogele 1994:9). This level of coding accuracy is comparable to that achieved in event data projects using human coders: Burgess and Lawton (1972:58) report a mean intercoder reliability of 82% for eight projects where that statistic is known.

In working with KEDS and later TABARI, we discovered additional advantages to machine coding. First, it is free of non-reproducible coding biases. Human coding is subject to systematic biases because of assumptions made by the coders. For example, Laurance (1990) notes that even expert coders at the U.S. Naval Postgraduate School tended to over-estimate the military capability of China because China was a large Communist country. Because most human event coding is done part-time by a rapidly changing group of students over a long period of time, coder biases are difficult to control. In contrast, when machine-coding is used, a set of words describing an activity will receive the same code irrespective of the actors or time period involved. Shallow parsing also tends to make *random* coding errors that a statistical analysis can rectify. in contrast, human coders tend to introduce systematic errors that are much more difficult to correct. Any biases embedded in the machine coding system are preserved *explicitly* in its vocabulary; there is no such record in human coding.

4 Tabari

KEDS was written in the PASCAL programming language and worked only on Apple Macintosh computers. The choice of PASCAL made sense at the time—it was the core language for the Macintosh operating system and my visceral loathing of Microsoft made the Macintosh the only option if I were to be doing the programming.[11] However, by the late 1990s PASCAL had been largely superceded by the C/C++ as the most common general-purpose programming language and compiler support for the language was dwindling. Furthermore, while KEDS was generally stable from 1995 to 2000, it contained some deep-seated idiosyncrasies that could only be eliminated by completely re-writing the program.

In response to this, TABARI —Textual Analysis By Augmented Replacement Instructions—was created in the spring of 2000.[12] It is based on the same shallow-parsing principles as KEDS but is written as "open-source" code in ANSI C++ and was immediately ported to the LINUX and WINDOWS operating systems. The conversion to C++ resulted in a program that was substantially faster than the PASCAL code—the program codes about 8,500 records per second even on an inexpensive machine, a $500 1.2 Ghz G4 Mac Mini. Upgrading to a 1.6 Ghz G5 gives a speed of

[11] Linux was still a gleam in Linus Torvald's eye when work began on KEDS, and machines running various flavors of UNIX were quite expensive. This is Kansas: we don't do expensive.

[12] "Tabari" is the name of an eminent 9th century Islamic historian Abu Jafar Muhammad ibn Jarir at-Tabari. I'd originally intended to name the program the Egyptian god of language, figuring this would give me a great logo as well. Unfortunately, while such a god exists, the god's name is "Thoth," which sounds very awkward to English speakers, and it is represented as having the head of an ibis or a baboon: not great logo material. Consequently TABARI, unlike KEDS, does not have a logo. KEDS used a tennis shoe as its logo, and this would start tapping as the program laboriously loaded dictionaries. The KEDS shoe corporation has never had much of a web presence, and consequently we still occasionally get email for them.

10,500 events/second. The coding speed on a Mini is about 300-times faster than KEDS, and about 33-million times faster than typical human coding.

In order to maximize compatibility between operating systems, TABARI uses a very simple UNIX-style keyboard-driven "terminal" interface rather than the GUI—Graphical User Interface—approach of KEDS. At one point we considered adding a GUI, but coders who have worked with both KEDS and TABARI consistently prefer the simpler interface, which is quite fast once one has memorized a few simple menus. The interface is implemented using the UNIX "ncurses" terminal library, and consequently we now have 100% compatibility between the Macintosh and UNIX/LINUX versions, as well as allowing the program to run remotely from a server.[13]

By re-writing the program from the ground up, TABARI eliminated an assortment of pathological coding behaviors in KEDS, while introducing relatively few new ones. TABARI incorporated several grammatical features that had been missing from KEDS, such as attempting to disambiguate verbs that can also be nouns (e.g. ATTACK and FORCE) by detecting the presence of the articles A, AN, and THE and mid-sentence capitalization, and providing considerably more sophisticated handling of compound phrases.

TABARI is generally stable and has been used intensively in a number of projects at the University of Kansas Center for International Political Analysis over the past five years. We have periodically added some additional specialized features at the request of external funders, and are slowly eliminating a few remaining quirks, but for the most part are no longer doing active development. Instead, we are producing data sets, which was the reason we got into this project in the first place. Fifteen years ago.

5 CAMEO

The most recent development in our project has been the CAMEO—Conflict and Management Event Observations—coding scheme. This is a new coding system specifically designed for automated coding, and has also evolved to accommodate the post-Cold War emphasis on political events involving sub-state actors. This work has been primarily done by my collaborator Deborah Gerner and her graduate student Ömür Yilmaz.

CAMEO came out of an NSF project where we [gasp!] decided to shift to a substantive focus—conflict mediation—rather than simply producing data. We figured we could construct and implement a new coding scheme in about six months. As usually happens with these things, the process actually took about three years, albeit in part this was due to interruptions by contract funding opportunities that were not always directly related to CAMEO, and various other complications.

CAMEO had three objectives. First, it was supposed to provide substantial detail on a specific type of political activity, and we originally envisioned it as an example of how automated coding could produce data customized for a specific theoretical problem, in contrast to the one-size-fits-all approach of WEIS and COPDAB. This development, of course, would be done entirely by modifying the dictionaries: the recoding itself would only take a couple of minutes.

Second, we wanted to produce a system that would resolve some long-standing ambiguities, inconsistencies and omissions in the 1960s coding systems. In this respect, we were engaged an effort similar to IDEA. In contrast to IDEA—which has carefully maintained backwards compatibility with all of the major earlier coding systems—we were willing to simply throw out or combine categories as needed.

Finally, CAMEO is specifically designed for machine coding. The era of human coding has ended (except for small specialized data sets), so if a distinction cannot be made using automated methods, we don't make it.[14] This has resulted in several of the WEIS cue categories being merged.

[13] We've had less success finding someone to keep the WINDOWS version current. Whatever...

[14] While it nominally maintains backwards compatibility, this must also effectively be the case for IDEA. Unlike TABARI, where the program and the dictionaries are distinct, the VRA CODER is apparently tightly linked to the IDEA coding system. Unfortunately, the program is proprietary and few details concerning its internal structure are available in the literature, so one cannot determine the details of this. VRA CODER has a substantially more sophisticated syntactic engine than TABARI —it is apparently a full parser—and consequently able to make a greater

As the CAMEO project developed, two additional features emerged. First—quite to our surprise—we ended up eliminating a number of WEIS categories, as well as categories which we had initially added because we were unable to find examples of the events in the news wire texts! Our manual `http://www.ku.edu/~keds/CAMEO.dir/CAMEO.v1.codebook.pdf` provides illustrative examples of texts corresponding to each coding category, and for some categories we could not find a single unambiguous example despite searching tens of thousands of news reports. These codes were, appropriately, eliminated from the system. This process of finding examples for the manual also uncovered numerous ambiguities and inconsistencies we had not previously considered, and was one of the reasons that the development of the system took such a long time.

The second feature emerged out of our contract work: the need to standardize actor coding as well as event coding. As we coded increasingly diverse sets of post-Cold War conflicts, we realized the need for a systematic method of coding new sub-state actors so that we could allow comparison across data sets (for example using the same conventions for distinguishing between militarized and non-militarized opposition groups), and to be able to establish rules for determining new codes (for example for the emergence of a new political organization) without having to call a meeting. At the request of one of our sponsors, we also switched from the older nation-state codes developed by the political science community in the 1960s to internationally recognized codes from the United Nations and International Standards Organization (ISO) whenever possible.

As a consequence, CAMEO—once intended as a very specific data coding framework—has evolved into something closer to a universal system, and incorporates an actor-coding standard absent in earlier efforts. We have gradually converted our primary working dictionaries to CAMEO—another process that took longer than anticipated, as did merging the work done on dictionaries developed for different geographical regions—and expect to do all of our future work only in CAMEO.

6 Dictionary Development

Most of the value-added from the KEDS project has been provided by the twenty or so coders[15] who have been involved with the project and devoted thousands of hours to refining the dictionaries that are essential to producing data. Without this effort, KEDS and TABARI would merely have been exercises in computer programming. As I am endowed with the inter-personal skills typical of a computer programmer, this aspect of the project has been managed by Deborah J. Gerner.

Dictionary development requires a great deal of skill and training. Fortunately, compared to most data-generation processes in the social sciences, it is relatively interesting, presenting an ever changing set of puzzles. For the right person, it is an attractive job: in contrast to many projects we are acquainted with, we have *very* low turn-over in personnel. A small number of coders are *really* good and at times were able to describe program bugs in sufficient detail that I knew exactly where in the code to make the change. An eloquent description of the challenges of dictionary development can be found in Joseph Pull's "Ode on Coding" `http://www.ku.edu/~keds/home.dir.ode.html` which Pull wrote prior to leaving the project for Yale Law School.

We have recruited coders almost exclusively from KU's undergraduate honors program, in many cases from students we have identified in classes we have taught. In most cases, the students are political science or international studies majors, though we have never made this a pre-requisite. Beyond that, we've had a difficult time figuring out what does or does not make a good coder. One of the smartest and most politically-savvy undergraduates we've known—someone we were certain would be perfect for the job—turned out to be absolutely incapable of coding (we found other tasks for her...). When we hire a cohort of six new coders, we can figure that one or two will disappear after a month or two. The rest will stay around until graduation (or longer).

We have discovered that the best way to hire coders is to have our existing coders do much of the interviewing (with one faculty member and either the data manager or a graduate student

number of distinctions than TABARI, but in the end IDEA is still constrained by the capabilities of VRA CODER

[15] We consistently have used this term, although "dictionary developers" is probably more accurate.

also participating). The "millennium generation" of honors students have spent their short lifetimes perfecting their résumés: "High school senior honors project: Amnesty International delegate to the Bonzo district of Jerkmenistan; personally intervened to stop ethnic cleansing campaign against the Calabash minority." These tactics have been finely/expensively honed for the express purpose of conning potential Boomer employers and admissions committees, but fail miserably when subjected to the assessment of peers. Peer interviewing—with the occasional principal investigator over-ride—also increases the likelihood that the individuals can work as a team.

Once hired, coders typically go through a period of about six weeks (10 to 15 hours per week) of practice coding on sample texts before being unleashed on actual dictionary development. This begins with a day-long "coding camp" that provides both an overview of event data research in general, an introduction to TABARI, and an extended discussion of the event and actor coding schemes. After this, most of the training is done by peers: for this reason, we've always tried to make sure that we are training the next cohort of coders at least a semester before the previous cohort graduates. This strategy requires a steady flow of funding, which, as discussed below, we've been fortunate to have.

In the traditional structure of social science research projects, the undergraduates would be supervised by a graduate student. We have done this at some points, but in recent years we have come up with sufficient funding to hire a 30 to 40 hour-per-week "data manager," usually selected from among the graduating undergraduates. This provides both continuity and an individual who is acquainted with the nuances of learning to code as an undergraduate. Our data managers have usually worked one or two years before moving on to [some very highly ranked] law school. We also have had several graduate students associated with the project for multiple years, particularly in the KEDS phase of the project.

Our projects have generally been run using a teamwork model rather than a hierarchical command-and-control model. We typically have project meetings every two to three weeks where we review current tasks, keep students appraised of our ever-erratic funding situation, identify bugs or possible new features for TABARI, and review news story leads that seem problematic with respect to our coding schemes. "Senior staff" meetings involving only the principal investigators, data manager, and graduate students are held when appropriate. Staff turnover is reduced by the liberal applications of cookies, donuts, birthday cakes, 21st birthday celebrations at the local brewpub, and other inducements to attend meetings that, due to scheduling problems, are invariably either early in the morning or late in the afternoon. During most of the past ten years we benefited tremendously from the support of KU's Policy Research Institute, where after a period of mutual adjustment between our twenty-somethings and the Boomer econometric modelers of the PRI professional staff—at one point we jokingly referred to our operation as CNUUM: Center for Noisy Undergraduates using Macintoshes[16]—we have reached a mutual accommodation.

Another feature that came out of the contract work was the development of an increasingly diverse set of utility programs to support the production of event data. The most important of these have always been our automated downloading programs, which have evolved from a script running a dial-in connection followed by processing in PASCAL to an integrated PERL program that does downloading and reformatting from HTML files taken off the web. ACTOR_FILTER is another one of our stalwarts: this identifies the actors in a set of text based on capitalization patterns, and produces a keyword-in-context index of these, sorted by frequency. With this tool, the most common actors (or items that *look like* actors) can be identified systematically, rather than on an ad hoc basis.

7 Funding

The KEDS and TABARI systems—both the programming and the more labor intensive dictionary development—have been funded by a combination of NSF grants and government contracts, with occasional bridge funding from KU, and an interesting contract doing conflict monitoring for a

[16]PRI is a Windows shop. Whatever...

Swiss-based NGO. We've been lucky (okay, we've also made our luck...): money has always been available for the tasks we needed to do, and in fact at times we've turned down work because of our limited number of trained coders [take the hint: if you can master the relevant tools, there is more funding available for this than we can handle].

NSF funding has generally been straightforward (but yes, we've also occasionally had to do revise-and-resubmits). U.S. government funding is another story: this has tended to operate on the "hurry up and wait" model where a funder will dangle possible funding for months, and then when permission finally comes through, one has to respond within days (in one case, we needed to provide a fully-approved contract in 24-hours; in another we didn't actually receive the funding until after the project had been completed (!) due to difficulties the KU had in finalizing the terms of contract). KU's grants office manages to cope with this, so I suspect they are used to it. The other frustration with government funding is that unlike NSF, contracts specify the delivery of a specific product, not to provide one's scientific expertise, and at times we've had to complete projects despite it being very clear that these were dead-ends, whereas with NSF we would have changed tracks and done something more useful.

We've always kept our work unclassified, for both principled and pragmatic reasons. We've no desire to become the Kansas equivalent of Los Alamos scientist Wen Ho Lee and advance the career of a zealous FBI agent anxious to get out of Topeka.[17] Meanwhile the value-added of classified material in the real world doesn't quite live up to its portrayal in the movies: consider for example the timely warnings provided for the collapse of the Soviet Union, India nuclear weapons tests, Iraqi WMDs and the recent Hamas electoral victory.

The KEDS project has generally been a relatively small affair: typically Gerner and me, one or two graduate assistants, a data manager, and a half-dozen coders. We've been larger—last summer PRI's accountant came to me and said "Do you realize you've got twenty people on your payroll??" (uh, no, I hadn't—kinda creeps up on you...)—but smaller is the norm. I occasionally get emails from people wanting to come and visit "our shop"—presumably envisioning the vast KEDS Building with its own cafeteria, weight room and day-care center. I explain that there really isn't a shop, just a web page: nothing to see here, move along, move along.

I've done most of the programming, though I've been fortunate in hiring a couple of first-rate student programmers over the years. They tend to disappear after about six months. I've also hired some miserable programmers: they also disappear. I've concluded that I'm not really very good at managing programmers—I can barely manage myself—though given the failure rate of programming projects in general, neither can most people.

8 Mama don't your babies grow up to be event data analysts

At this point we have spent five years in initial experimentation with automated coding methods, devoted about fifteen years to operational program and dictionary development, produced regional data sets for about thirty countries, and now have the capability of maintaining data sets with a resolution of about a day at close to zero marginal cost (or at least a lower marginal cost than any other known method of creating data in the social sciences, including curb-stoning). Event data analysis has therefore taken the quantitative international relations world by storm, right?

Well, no. While articles utilizing event data have appeared on a relatively regular basis in all of the refereed "sacred journals" that carry quantitative work, it remains very much a niche approach in international relations and comparative politics.[18] Individuals focusing on event data analysis have, with a couple of exceptions, not fared particularly well in the academic job market: in fact the individual who I feel was doing some of the very best work outside of KU was, at last report, running a coffee shop.

Event data has fared substantially better in the policy community, and several people who have been unable to secure academic employment have gone on to positions as quantitative policy

[17] Trust us, any FBI agent based in Topeka will want to get out.

[18] In the contemporary conflict forecasting work we do, these two fields are essentially indistinguishable.

analysts in the defense and intelligence communities. There they pull down salaries twice those of academics, don't have to attend faculty meetings, don't grade bluebooks, and can't take their work home because it is classified.

This latter point, however, means that we have had very little feedback from the policy community. Six months after one of my best-trained students took a defense-related job where he was hired explicitly for his event data training, we met at the APSA. "Job going well?" says I. "Yep." says he. "Bet you can't tell me a single thing about it." says I. "Yep." says he. Consequently we can only infer how much the methods are being used by questions and requests for data (and on quite a few occasions, putting people in one part of the U.S. government in touch with other people in the U.S. government who are doing the same thing).[19]

What we do know from our own tests is that forecasting models using event data work: at policy-relevant lead times of 1 to 6 months, even very simple models can predict high conflict levels in well-covered protracted conflicts (for example Israel-Palestine, Israel-Lebanon, and Serbia-Bosnia) with 70% to 80% accuracy in full-sample assessments, and 60% to 70% in out-of-sample assessments. Based on Tetlock's (2005) recent work, this is probably substantially better than human analysts. If this is the level of accuracy for simple models, additional systematic research presumably could push the accuracy even higher.

Three things stand in the way of this. The first is paradigmatic: quantitative research in international relations is dominated by the "Correlates of War" approach that has almost nothing in common with event data analysis. COW studies typically involve the analysis of interval-level variables measured at the nation-state dyad-year level across two centuries and the entire international system. In contrast, contemporary event data analysis focuses nominal measurements in protracted conflicts across a couple of decades or less, but with daily resolution and an increasing focus on sub-state actors. The COW community has generally focused on retrospective inference guided by theoretical issues (initial balance of power theories, more recently the democratic peace); the event data community on policy-relevant forecasting. There is virtually no overlap in the theoretical questions, measurement issues, or statistical techniques.

The second problem involves the shortage of nominal-level time series methods. These exist—for example hidden Markov models—but they are generally closer to pattern recognition methods than to classical frequentist statistics. Interval-level time series are used extensively in econometrics, a field already familiar to most political methodologist. In contrast, the two major sources of nominal-level methods are the very unfamiliar fields of linguistics and bioinfomatics. The latter wasn't even recognized as a distinct field until about a decade ago, when improved DNA sequencing techniques made vast amounts of data available. Of course, now that it is almost as easy to lose large amounts of money with bad nominal-level methods in biotechnology as it is to lose large amounts of money with bad interval-level methods in the stock market, we are likely to see further developments.

The third issue is the complexity of the data generation process. I have a detailed, step-by-step list of the steps requires to get from NEXIS to a data set that can be used in a statistical or pattern-recognition model, and it involves something like 60 operations and the use of about a half-dozen customized utility programs. We've been able to make this process somewhat smoother over the years, but the under the pressure of producing data, working ad hoc solutions tend to predominate over elegant user-friendly software. It is possible to do this independently (familiarity with C and PERL help), but it is a whole lot more complicated than simply downloading some pre-compiled data set off the web.

Central to the Japanese "manufacturing miracle" of the second half of the twentieth century was the concept of *kaizen*—incremental improvement. A worker's suggestion that increases the quality of a product by only 0.1% will, when combined with similar suggestions by thousands of workers over a period of decades, provide the technological leverage to reduce a device for playing music from the size of a suitcase to the size of a pocket knife, while hugely increasing capacity and quality.

[19]This "I could tell you but then I'd have to kill you" problem has also affected forecasting models using rational choice methods. These may, or may not, be extensively used in the intelligence community, depending on who you want to believe.

TABARI and the VRA CODER are improvements over KEDS, and TABARI can be incrementally augmented through the open-source development process. The CAMEO and IDEA coding schemes are improvements over WEIS and COPDAB. Each time a coder finds another verb phrase to add to the dictionary, or adds another name to the list of actors being coded, the probability of sentences being coded correctly increases, however slightly. Those improvements in the coding software, the coding schemes, and dictionaries are preserved, rather than disappearing when a cohort of coders graduates.

At this point we probably have a good idea of how to produce event data—all event data articles published in major journals over the past ten years have used machine-coded data, and the last major human coding project was shut down in 2004 following a comparison between its data and a comparable data set produced using TABARI. We still need to take the next step in figuring out some really good things to do with it. But at least we've started.

9 For further information:

The KEDS project maintains a very extensive web site at `http://www.ku.edu/~keds` At this site you will find the most recent versions of the software and documentation, assorted coding dictionaries, data sets and utility programs, a FAQ (frequently-asked-questions) section, and copies of papers from our project and related efforts.

10 references

Advanced Research Projects Agency (ARPA). 1993. *Proceedings of the Fifth Message Understanding Conference (MUC-5).* Los Altos,CA: Morgan Kaufmann.

Azar, Edward E. 1980. "The Conflict and Peace Data Bank (COPDAB) Project." *Journal of Conflict Resolution* 24:143-152.

Bond, Doug, Bennett, Brad and Vogele, William. 1994. Data development and interaction events analysis using KEDS/PANDA: an interim report. International Studies Association, Washington.

Bond, Doug, J. Craig Jenkins, Charles L. Taylor and Kurt Schock. 1997. Mapping Mass Political Conflict and Civil Society: The Automated Development of Event Data. *Journal of Conflict Resolution* 41, 4: 553-579.

Burgess, Philip M. and Raymond W. Lawton. 1972. *Indicators of International Behavior: An Assessment of Events Data Research.* Beverly Hills: Sage Publications.

Gerner, Deborah J., Philip A. Schrodt, Ronald A. Francisco, and Judith L. Weddle. 1994. "The Machine Coding of Events from Regional and International Sources," *International Studies Quarterly* 38:91-119.

Goldstein, Joshua S. 1992. "A Conflict-Cooperation Scale for WEIS Events Data." *Journal of Conflict Resolution* 36: 369-385.

Hudson, Valerie, ed. 1991. *Artificial Intelligence and International Politics.* Boulder: Westview

King, Gary and Will Lowe. "An Automated Information Extraction Tool For International Conflict Data with Performance as Good as Human Coders: A Rare Events Evaluation Design" *International Organization* 57,3: 617-642.

Laurance, Edward J. 1990. "Events Data and Policy Analysis." *Policy Sciences* 23:111-132.

McClelland, Charles A. 1976. *World Event/Interaction Survey Codebook.* (ICPSR 5211). Ann Arbor: Inter-University Consortium for Political and Social Research.

Merritt, Richard L., Robert G. Muncaster and Dina A. Zinnes, eds. 1993. *International Event Data Developments: DDIR Phase II.* Ann Arbor: University of Michigan Press.

Schrodt, Philip A. and David Leibsohn. 1985. "An Algorithm for the Classification of WEIS Events from WEIS Textual Data." Paper presented at the International Studies Association, Washington, March 1985.

Schrodt, Philip A. and Alex Mintz. 1988."A Conditional Probability Analysis of Regional Interactions in the Middle East." American Journal of Political Science. 32,1: 217-230.

Schrodt, Philip A. and Deborah J. Gerner. 1994. "Validity assessment of a machine-coded event data set for the Middle East, 1982-1992." *American Journal of Political Science* 38: 825-854.

Tetlock, Philip E. 2005. *Expert Political Judgment: How Good is It? How Can We Know?* Princeton: Princeton University Press.

Trappl, Robert, ed. 2006. *Programming for Peace : Computer-Aided Methods for International Conflict Resolution and Prevention.* Berlin: Springer.