Coding Negotiations with AI: Instructions and Validation for Coding Model 3

NTR AI Negotiation Coder

Ray Friedman[a], Xuhui Zhan [a], Shivam Tyagi[a], Jeanne Brett [b],
Madison Hooper[a], Katie Babbit [a], Manish Acharya[a]

[a] Vanderbilt University
[b] Negotiation and Team Resources (NTR)

April 28, 2026

ABSTRACT

This paper provides guidance for scholars wanting to use AI negotiation transcript coding Model 3 of the NTR AI Negotiation Coder. This includes a link to the Lab's website, a description of Model 3's codes and coding process, and a report of tests that support the model's validity. It also includes instructions for how to set up and submit your data. When reporting your results from this model, please cite this paper.

Introduction

The NTR AI Negotiation Coder provides AI-assisted coding of negotiation transcripts. The site provides access to several coding models. For each model, the site provides: definitions of the codes, examples of each code, details of how the AI model was developed and validated, and instructions for how to format and submit your transcripts for coding. That information is also contained in this document.

These models were trained in most cases by learning from transcripts that have been human coded for specific prior projects (a full review of the development of this coding process can be found in Friedman et al, 2024). Each project used a different coding scheme, and even when specific codes were the same (e.g., "substantiation"), that code may have been interpreted slightly differently in different projects. The example sentences provided on the website allow you to see how each project used its codes, so you can decide which model fits your research needs.

We plan to add more coding models and to update models as needed. If you have a set of coded transcripts for a coding scheme that you would like us to include, please contact us.

Model 3 – Kern, Brett, Weingart, and Eck (2020)

Model 3 uses the coding scheme of Kern, Brett, Weingart, and Eck (2020), "The 'fixed' pie perception and strategy in dyadic versus multiparty negotiations." Their coding scheme began with 37 codes but was later reduced to four strategic categories for their analysis. Kern et al (2020) describe their coding process:

> *[We] coded each speaking turn using a 37-category coding scheme based on one published by Weingart et al (2007). … A two-dimensional correspondence analysis (Greenacre, 1993) aggregated the 37 codes into four categories with results generally replicating Weingart et al.'s (2007) clusters of strategic orientation: integrative versus distributive, and of strategic function: information seeking and sharing versus reaching agreement*. (Kern, 2020, p. 148)

We retained a version of the more elaborate coding scheme and the simplified version of their coding scheme in. In our Model 3, we include two parts to the analysis: a more detailed version of the coding system (based on the 37 codes, reduced to 26 codes) and a shorter version (based on the four strategic clusters). Kern et al.'s (2020) Table 1 (partially reproduced here in Appendix 1) shows how the larger set of codes mapped onto the four broader "strategic" categories of: *value creating, integrative information, value claiming,* and *distributive information. (*Note that seven of the initial 37 codes were not included in any strategic category, for reasons described in Appendix 1.)

26 Code Output. We began with the 37 codes. Then we took out one code ("vote on one issue") because this code did not appear in any transcripts. We also realized that we had too many codes for the model to categorize at reasonable levels. We examined the initial confusion matrices and the codes themselves, looking for codes that were hard to distinguish, but which fell into the same strategic category. The reduction of 37 codes into 26 codes is shown in Appendix 2. One part of the output provided by our model applies these 26 codes to submitted transcripts.

5 Code Output. The second part of the output provided by our model is the four strategic categories. In this output, a given speaking turn is placed into one of the four strategic categories based on which of the 26 codes it is assigned, or "other" if the assigned code was not included in one of the categories by Kern et al. (2020). Note that this output is simply a transformation of the 26-code output.

The simulation used in the Kern et al.'s (2020) study was *Towers Market* (negotiationandteamresources.com). The authors shared their human-coded transcripts of this simulation. We only included in our training and testing sets the 32 transcripts that used the dyadic version that the simulation.

To understand the initial set of 37 codes see the coding manual used by Kern et al (2020). This document provides the codes, definitions, and examples of each code. In addition, to help you understand how these codes were applied by coders for Kern et al. (2020) we prepared a list of 10 sentences that had been assigned to each code (or fewer if there were less than 10 instances of that code in their transcripts). Those sentences are in the first column of this document.

When we fine-tuned the model, we assumed that effective fine-tuning would require at least 10 instances of a code to appear in the five training transcripts, but that did not always occur. In cases where there were less than 10 instances, we created additional sample sentences, shown in the second column of this document. These sentences were generated by asking Chat GPT4, which was given the existing small set of examples and the description of the code in the coding manual, to produce 15 new examples. These newly created sentences were checked for accuracy by the first author, with inappropriate examples deleted or modified. These examples were then added to the prompt.

The sample sentences (and extra examples we generated) let you see how these codes were operationalized in our model – this is what Model 3 learned from and tries to reproduce when coding your transcripts. As with any coding scheme, different scholars might operationalize concepts slightly differently. You should decide if this coding scheme is useful to you by reviewing how the authors used it.

Our Fine-Tuning Process

This model uses Anthropic's Claude Sonnett 4.0 model, and in-context learning. This means that the base model was finetuned by the same prompt that asked it to analyze a new transcript. That prompt includes five coded transcripts from Kern et al. (2020), selected to optimize the coverage of codes, as well as the additional sentences that we created, definitions of the codes, instructions for how to code, and instructions for how the output should be formatted.

Coding Process

To use Model 3, your transcripts need to be coded in three steps.

1. **Unitization (you need to do this).** The model provides one code for each set of words or sentences that you identify as a unit in your Excel document. You can choose to have units be speaking turn, sentences, or thought units. The easiest to set up is speaking turns, since switching between speakers is clearly identifiable in transcripts. The next easiest is sentences, since they are identified by one of these symbols: .?! However, different transcribers may end sentences in different places. The hardest unit to create is the thought unit since that takes careful analysis and can represent as much work as the coding itself. (See the NegotiAct coding manual for how to create thought units; Jackel et al., 2022). Clarity of meaning runs in the opposite direction. The longer the unit, the more likely there are multiple ideas in the unit, and less clarity for human or AI coders to know what part to code. Kern et al. (2020) coded speaking turns, but 92% of their speaking turns contained just one sentence. The closest alignment with the training data would be for you to use sentences as the unit.

2. **Model Assigns Codes.** The model assigns a code to each unit you submit, based on in-context learning. Coding is guided by the prompt we developed and tested. For more about in-context learning see Xie and Min (2022). Our prompt for this model includes several elements:
    a. Five fully coded transcripts. These transcripts were chosen from the 32 available transcripts in the following way. First, any combination of five was considered only if that set included all 35 codes. Second, one combination was chosen at random to test.
    b. Instructions to pay attention to who was speaking, such as "buyer" or "seller".
    c. Instructions to pay attention to what was said in the conversation before and after the unit being coded.
    d. Additional examples of any codes (as discussed above) when the five training transcripts did not contain at least 10 examples.

3. **We Run Model Five Times.** We automatically run the model five times, to assess consistency of results. As expected, the results are not always the same, since with in-context learning the model learns anew with each run and may learn slightly differently each time. Variation is also expected since - some units may reasonably be coded in several ways. By running the coding model five times, we get five codes assigned to each speaking unit. If three, four, or five of the five of the runs have the same code, we report the code and indicate the level of

"consistency" of that code (three, four, or five out of five). If there are not at least three consistent results out of five runs, or if the model fails to assign a code, we do not report a model code. In these cases, the researcher needs to do human coding.

Validation of Fine-Tuned Model

This section reports the initial validation, using Claude Sonnet 4.0. This version of Sonnet was deprecated on April 14, 2026, and replaced with Sonnet 4.6. Updated validation of key measures is reported at the end of this section and shows equivalent results for Sonnet 4.6. Validation occurred in several steps.

Step 1: **Compare the Model Coding with Human Coders for Kern et al. (2020).**

We asked the model to code the 2,970 units contained in the Kern et al. (2020) transcripts that were not selected for training, and evaluated the results in two ways:

1. **Consistency.**
    a. *26 Code Version*. In our test, there was "perfect" consistency of model coding (five out of five runs of the model assigned the same code to a speaking unit) for 2922 of the units, "high" consistency (four out of five) for 29 of the units, "modest" consistency (three out of five) for 19 three of the units, and none where the model did not report a code. The results showed perfect consistency for 98.38% of codes.
    b. *5 Code Version*. In our test, there was "perfect" consistency of model coding (five out of five runs of the model assigned the same code to a unit) for 2931 of the units, "high" consistency (four out of five) for 24 of the units, "modest" consistency (three out of five) for 15 of the units, and 0 cases where the model did not report a code. The results showed perfect consistency for 98.69% of codes.

2. **Match with Human Coding.**
    a. *26 Code Version.* We assessed whether the model assigned the same code as the human coders. The overall match level for units where the model assigned a code (i.e., where at least three of the five runs assigned the same code) was 74.18% (see Table 1a).
    b. *5 Code Version.* The overall match level for units where the model assigned a code (i.e., where at least three of the five runs assigned the same code) was 77.81% (see Table 1b).

Table 1a: Match Percentage by Consistency Level, Validation Step 1– 26 Code Version

Level of Consistency	Match with Human Codes	% Achieve This Consistency Level Among Those Assigned a Code	Number	Match	Percentage Match
Modest Consistency	3 out of 5	0.64%	8	not match	
			11	match	57.89%
High Consistency	4 out of 5	0.98%	19	not match	
			10	match	34.48%
Perfect Consistency	5 out of 5	98.38%	740	not match	
			2182	match	74.67%

*0 cases did not reach the 3 out of 5 consistency threshold or the model failed to assign a code

Table 1b: Match Percentage by Consistency Level, Validation Step 1 – 5 Code Version

Level of Consistency	Match with Human Codes	% Achieve This Consistency Level Among Those Assigned a Code	Number	Match	Percentage Match
Modest Consistency	3 out of 5	.51%	7	not match	
			8	match	53.33%
High Consistency	4 out of 5	0.81%	11	not match	
			13	match	54.17%
Perfect Consistency	5 out of 5	98.69%	641	not match	
			2290	match	78.13%

*0 cases did not reach the 3 out of 5 consistency threshold or the model failed to assign a code

We also calculated Cohen's kappa, with the model codes as coming from one rater and the human coding as coming from a second rater. This calculation, compared to the "percentage match," accounts for matches that might occur based on chance. Cohen's kappa was calculated in R (R Core Team, 2022) using the IRR package (Gamer & Lemon, 2019).

a. *26 Code Version*. Cohen's kappa was .70, with the no information rate of .21 (p-value of difference is < .001). According to Landis and Koch (1977) this represents "substantial agreement", and according to Fleiss (1981) is "fair to good" agreement. Rather than relying on conventional categorical guidelines to interpret the magnitude of kappa, Bakeman (2023) argues that researchers should estimate observer accuracy or how accurate simulated observers need to be to produce a given value of kappa. he KappaAcc program (Bakeman, 2022) was used to estimate observer accuracy, which was 85%.
b. *5 Code Version*. Cohen's kappa was 0.71, with the no information rate of .26 (p-value of difference is < .001). According to Landis and Koch (1977) this represents "substantial

agreement", and according to Fleiss (1981) is "fair to good" agreement. Rather than relying on conventional categorical guidelines to interpret the magnitude of kappa, Bakeman (2023) argues that researchers should estimate observer accuracy or how accurate simulated observers need to be to produce a given value of kappa. The KappaAcc program (Bakeman, 2022) was used to estimate observer accuracy, which was 87%.

It is also worth noting that in many cases where scholars establish inter-coder reliability, there is a process of cross-rater discussion that is used to resolve initial differences of opinion between the two coders. In a study of inter-coder agreement, initial coder agreements in the 80% range began with coder agreement in the 40% range (Garrison, et al., 2005). Of course, in our case there can be no cross-rater discussion between a model and a human, taking away one step that is often used to achieve higher kappas. The closest we can get to that process is to have a third person view the cases of human-model disagreement to provide a judgment of which code was more correct. Also, the fact that so many codes need human-to-human discussions to resolve, suggests some inherent ambiguity about code assignments and opens up the possibility that several different codes might reasonably be assigned to some segments of transcripts.

Confusion Matrix and Summary Data. We created a confusion matrix for all codes with modest, high, or perfect consistency (see Tables 2a and 2b). The vertical axis shows human coding. The horizontal axis shows model coding. We also included (see Tables 3a and 3b) summary statistics showing which codes appeared most frequently, and which had the highest levels of human-model match.

26 Code Version. Table 3a shows which codes appeared most frequently in the human coding for 26 codes. Substantiation was most common representing 26% of the codes, while QB (Questions about the Bottom Line) was least common representing just .03% of the codes. There appears to be a rough correlation between number of units and match percentage, suggesting that match percentage goes up when there are more examples of a code in the training transcript for the model to learn from and when there are more opportunities to find that code in the test transcripts.

5 Code Version. Table 3b shows which codes appeared most frequently in the human coding for 5 codes. Distributive Action was most common representing 33.8% of the codes, while Integrative Action was least common representing just 8.2% of the codes. The highest level of human-model match was for Integrative Information at 89% while the lowest was for Distributed Information at 65%.

Table 2a: Confusion Matrix, Validation Step 1 – 26 Code Version

AGREEMENT (AGR)
COERCION (COER)
CREATIVE SOLUTIONS (CS)
DIFFERENCES (DIFF)
DISAGREEMENT (DIS)
BOTTOM-LINE (IB)
INSIGHT (IN)
INTEGRATIVE PROCEDURE (INT_PROC)
ISSUE PREFERENCES (IP)
PRIORITIES (IR)
MISCELLANEOUS (MISC)
MUTUALITY (MU)
MULTI-ISSUE (OM)
SINGLE-ISSUE (OS)
DISCUSS ONE ISSUE (P1)
MOVING ON (PM)
TIME (PT)
ASK FOR BOTTOM LINE (QB)
ASK MISCELLANEOUS TASK RELATED QUESTIONS (QM)
ASK ABOUT OFFER (QO)
ASK FOR PREFERENCES (QP)
ASK FOR PRIORITIES (QR)
ASK ABOUT SUBSTANTIATION (QS)
DEFENDING ARGUMENTS (SB)
FACTUAL STATEMENTS (SF)
SIMILARITIES (SIM)

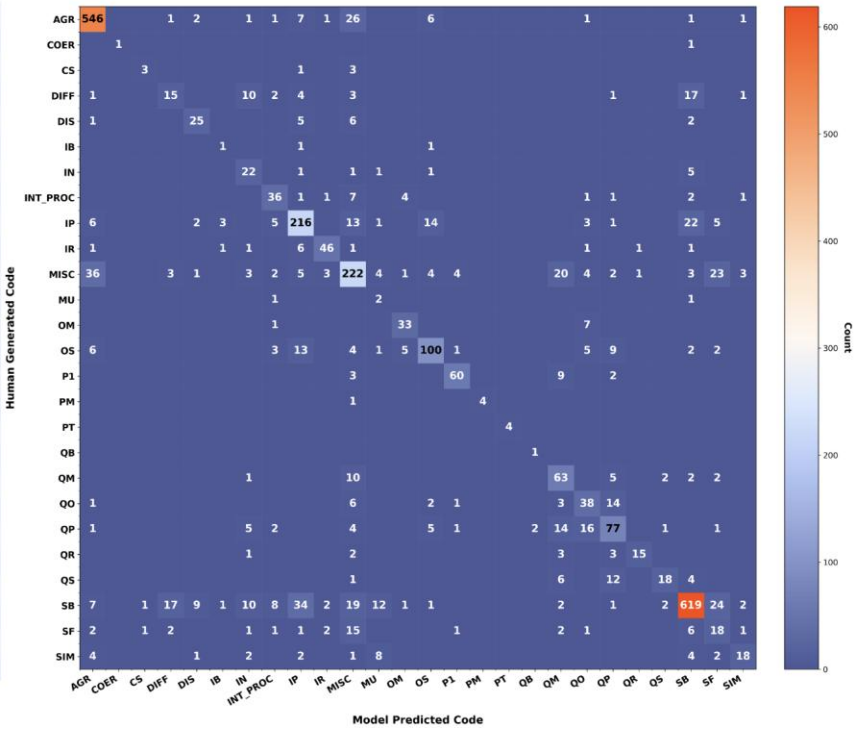

Table 2b: Confusion Matrix, Validation Step 1 – 5 Code Version

DISTRIBUTIVE ACTION (DA)
DISTRIBUTIVE INFORMATION (DI)
EXCLUDED (Excluded)
INTEGRATIVE ACTION (IA)
INTEGRATIVE INFORMATION (II)

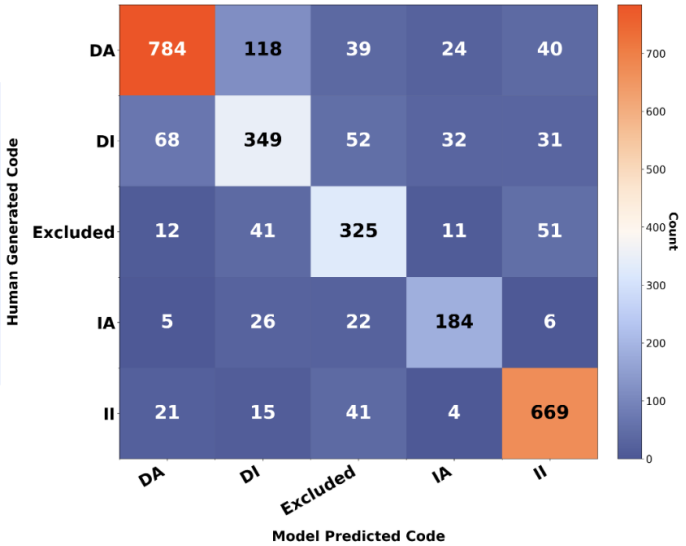

Table 3a: Match Percentage by Code, Validation Step 1, 26 Code Version

Human Code	% of units Across All Coded Transcripts	Model Match %
SB	25.99	80.18
AGR	20.00	91.92
MISC	11.58	68.53
IP	9.80	74.23
OS	5.08	66.23
QP	4.34	59.69
QM	2.86	74.12
P1	2.49	81.08
QO	2.19	58.46
IR	1.99	77.97
INT_PROC	1.82	66.67
DIFF	1.82	27.78
SF	1.82	33.33
SIM	1.41	42.86
OM	1.38	80.49
QS	1.38	43.90
DIS	1.31	64.10
IN	1.04	70.97
QR	0.81	62.50
CS	0.24	42.86
PM	0.17	80.00
PT	0.13	100
MU	0.13	50.00
IB	0.10	33.33
COER	0.07	50.00
QB	0.03	100

Table 3b: Match Percentage by Code, Validation Step 1, 5 Code Version

Human Code	% of units Across All Transcripts	Model Match %
Distributive Action	33.84	78.01
Integrative Information	25.25	89.20
Distributive Information	17.91	65.60
Excluded	14.81	73.86
Integrative Action	8.18	75.72

Closer Look at Mismatches: Mismatch Analysis.

To assess the nature of these and other mismatches, we selected a random sample of 120 mismatches for closer examination. Given that original human coders may be just as likely to make errors (or simply vary in their judgments) as the model, we wanted to see if newly trained coders would see the Kern-provided codes or the model-provided codes as more accurate. We provided these two coders with 120 speaking turns to categorize, as well as the two speaking turns preceding that speaking turn, along with the human and model codes. Coders were not informed which code came from the model or

humans, and the order in which the human and model codes were shown to them was reversed every 10 to 15 speaking turns, to avoid order effects. Coders selected which code they saw as more accurate. This was done first separately by the two coders, and then they resolved their differences through discussion.

In the end, these new human coders thought the model-provided codes were correct 57% of the time and the human-generated codes 36% of the time. In addition, in 8% of cases the coders either could not agree which code (human or AI) was right or thought that a third code was right. We labeled these 8% as "uncertain." Based on this analysis, we can expect that the model is correct in at least 57% of the cases with mismatches. This implies that we can trust that about 93.5% of the model's codes for the 26-code scheme are correct. That is, with an estimated observer accuracy of 85%, we expect that 15% are not correct. Of that 15%, 57% were deemed to be cases where the model was correct. This results (85% + [.57 x 15%]) in an estimate of 93.5% model accuracy.

Step 2: **Match with Human coding for Different Simulations**

The first step of validation involved matching human and model codes where the negotiation simulation used for training was the same as the negotiation simulation used for testing the model (*Towers Market*). But users may have transcripts from any number of simulations or real-world negotiations, not just the simulation used in the Kern et al. (2020) study. Therefore, we wanted to test how well the model would match human coders who applied the Kern et al. model to transcripts using other simulations. We selected a set of 3 transcripts from a study that used the Cartoon simulation, and 1 transcript from a study that used the Les Florets simulation. Since these transcripts were not initially coded using the Kern codes, we trained two coders to use the Kern codes. After initial training, they reached a level of inter-coder reliability of k=.75 on the first transcript. They coded the transcripts separately and came together to discuss any cases where they disagreed and assign a code. This provided the human coded transcripts from the Cartoon and Les Florets simulations. These transcripts were then coded using our model.

a. **26 Code Version:** The 4 transcripts had 755 speaking turns. The model had perfect consistency for 94.97% of the speaking turns, high consistency for 4.24% of the speaking turns, and modest consistency for 0.79% of the speaking turns. There were 0 cases of less than 3 out of 5 consistency. The match percentage was 55.65% for perfect consistency codes and 15.62% for high consistency codes and 16.67% for modest consistency (see Table 4a). Overall, the match percentage was 55.10%. This was lower than our prior tests, as expected, because these transcripts did not have the same issues and topics as the training transcripts (which used the Towers Market simulation). For that reason, these results may better represent the model's effectiveness with most transcripts.
b. **5 Code Version:** The 4 transcripts with 755 speaking turns also had 96.16% coded by the model with perfect consistency, 3.05% with high consistency, 79% with modest consistency. The match percentage for the perfect consistency was 62.12%, for high consistency 17.39% and for modest consistency 33.33% (see Table 4b). Overall, the match percentage was 60.53%.

We also calculated Cohen's kappa. For the 26-code version the weighted Cohen's kappa was .49 with the no information rate of .29 (*p*-value of difference is < .001), while for the 5-code version the weighted Cohen's kappa was .49 with a no-information rate of .36 (p-value of difference is <.001).

Rather than relying on conventional categorical guidelines to interpret the magnitude of kappa, Bakeman (2023) argues that researchers should estimate observer accuracy or how accurate simulated observers need to be to produce a given value of kappa. The KappaAcc program (Bakeman, 2022) was used to estimate observer accuracy, which was found to be .72 for the 26-code version and .76 for the 5-code version.

Table 4a: Match Percentage by Consistency Level, Validation Step 2 – 26 Code Version

Level of Consistency	Match with Human Codes	% Achieve This Consistency Level Among Those Assigned a Code	Number	Match	Percentage Match
Modest Consistency	3 out of 5	0.79%	5 1	not match match	16.67%
High Consistency	4 out of 5	04.24%	27 5	not match match	15.62%
Perfect Consistency	5 out of 5	94.97%	318 399	not match match	55.65 %

*0 cases did not reach the 3 out of 5 consistency threshold or the model failed to assign a code

Table 4b: Match Percentage by Consistency Level, Validation Stage 2 – 5 Code Version

Level of Consistency	Match with Human Codes	% Achieve This Consistency Level Among Those Assigned a Code	Number	Match	Percentage Match
Modest Consistency	3 out of 5	0.79%	4 2	not match match	33.33%
High Consistency	4 out of 5	03.05%	19 4	not match match	17.39%
Perfect Consistency	5 out of 5	96.16%	275 451	not match match	62.12%

*0 cases did not reach the 3 out of 5 consistency threshold or the model failed to assign a code

The proportion of speaking units that fell into each category were roughly similar to what we saw in the first validation tests, with most speaking units being: Agreement (AGR) followed by Miscellaneous (MISC). In this set of transcripts Substantiation was also fairly common (see Table 6). As with the first validation test, model-human match percentage appears to be correlated with number of codes.

The confusion matrix (see Table 5) shows that, once again, the largest number of mismatches comes from Agreement and Miscellaneous. It also shows that nearly all the mismatches were cases where the model is likely confused between the two overrepresented codes.

Table 5a: Confusion Matrix, Validation Step 2, 26 Code Version

AGREEMENT (AGR)
COERCION (COER)
CREATIVE SOLUTIONS (CS)
DIFFERENCES (DIFF)
DISAGREEMENT (DIS)
BOTTOM-LINE (IB)
INSIGHT (IN)
INTEGRATIVE PROCEDURE (INT_PROC)
ISSUE PREFERENCES (IP)
PRIORITIES (IR)
MISCELLANEOUS (MISC)
MUTUALITY (MU)
MULTI-ISSUE (OM)
SINGLE-ISSUE (OS)
DISCUSS ONE ISSUE (P1)
MOVING ON (PM)
TIME (PT)
ASK FOR BOTTOM LINE (QB)
ASK MISCELLANEOUS TASK RELATED QUESTIONS (QM)
ASK ABOUT OFFER (QO)
ASK FOR PREFERENCES (QP)
ASK FOR PRIORITIES (QR)
ASK ABOUT SUBSTANTIATION (QS)
DEFENDING ARGUMENTS (SB)
FACTUAL STATEMENTS (SF)
SIMILARITIES (SIM)

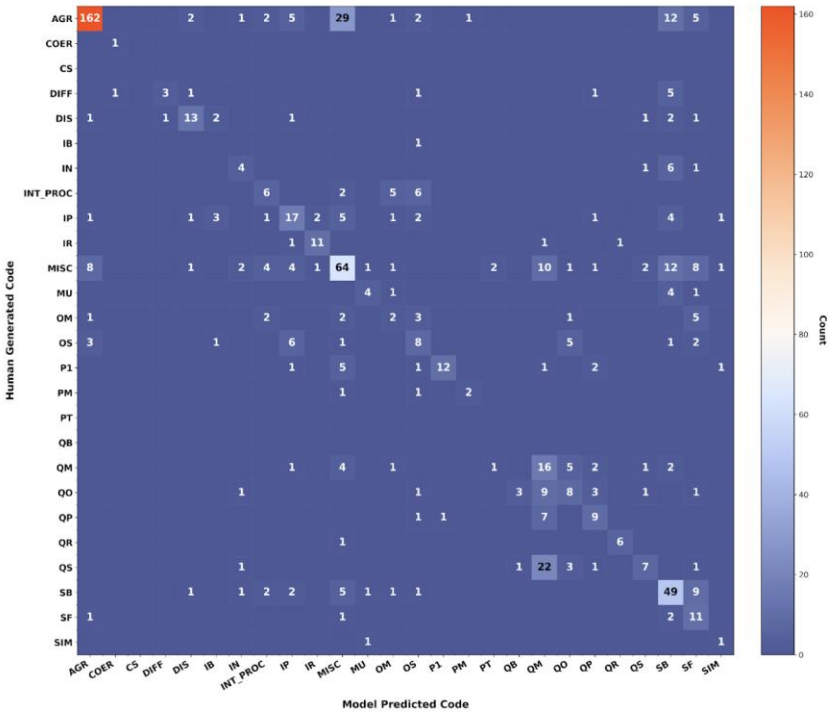

Table 5b: Confusion Matrix, Validation Step 2, 5 Code Version (UPDATE NEEDED)

DISTRIBUTIVE ACTION (DA)
DISTRIBUTIVE INFORMATION (DI)
EXCLUDED (Excluded)
INTEGRATIVE ACTION (IA)
INTEGRATIVE INFORMATION (II)

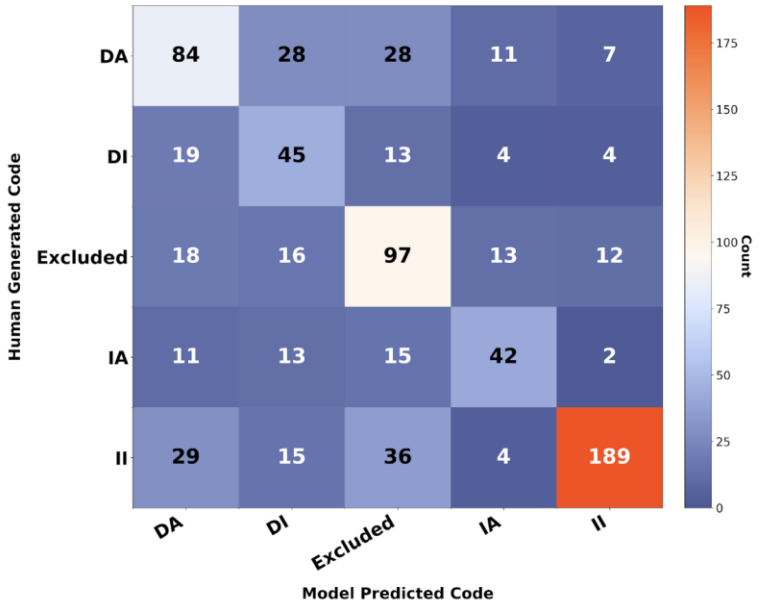

Table 6a: Mach Percentage by Code, Validation Stage 2 – 26 Code Version

Human Code	% of units Across All Transcripts	Model Match %
AGR	29.40%	72.97%
MISC	16.29%	52.03%
SB	9.54%	68.06%
IP	5.17%	43.59%
QS	4.77%	19.44%
QM	4.37%	48.48%
QO	3.58%	28.00%
P1	3.05%	52.17%
DIS	2.91%	59.09%
INT_PROC	2.52%	31.58%
QP	2.38%	50.00%
OM	2.12%	00.00%
SF	1.99%	73.33%
IR	1.85%	78.57%
DIFF	1.59%	25.00%
IN	1.59%	33.33%
MU	1.32%	40.00%
QR	0.93%	85.71%
PM	0.53%	50.00%
SIM	0.26%	50.00%
COER	0.13%	100%
IB	0.13%	00.00%

Table 6b: Mach Percentage by Code, Validation Stage 2 – 5 Code Version

Human Code	% of units Across All Transcripts	Model Match %
Integrative Information	36.16%	69.23%
Distributive Action	20.93%	53.16%
Excluded	20.66%	62.18%
Distributive Information	11.26%	52.94%
Integrative Action	10.99%	50.60%

Closer Look at Mismatches: Mismatch Analysis.

In order to analyze the mismatches, we collected a random sample of 100 sentences with mismatches from the 26-code results, along with the two prior and two following sentences (to understand context) and the human and model codes assigned to the coded sentence. We then took off the column labels and randomly mixed the order of the codes. Since the human coding in this case was done by our coding team, we wanted a different person to select which of the two codes was more correct. This was done by the first author. In 55 of the 100 cases, the model was deemed correct. In 17 cases, both the human and the model codes were deemed correct. This happened mostly due to having sentences where one part of the sentence could accurately be given the model code, while another part of the sentence could accurately be given the human code. For example, there were cases where a difference was expressed (DIFF) that served to substantiate an argument (SB) so both DIFF and SB could be considered accurate. In these 17 cases, we consider the model's code assignment to be correct, bringing the total for model correct to 72 of the 100 cases, or 72%. In 23 of the 100 cases, the human coders were deemed correct, while the model code was deemed incorrect. In the remaining five cases, both the human and model codes were deemed incorrect.

We started with a model estimated observer accuracy of 72%[1] for the 26-code version, suggesting a model estimated observer inaccuracy of 28%. Since the mismatch analysis shows that 72% of the time, we can estimate that the model was right when there was a model/human mismatch (72% of 28% is 20%), we can estimate that the model accuracy was 72% plus 20%, or 92%.

Confirm validation of model using Claude Sonnet 4.6 (replaced Sonnet 4.0 in April, 2026)

Here we report re-runs of the main analyses which show that the Sonnet 4.6 validation results are consistent with the validation results reported above for Sonnet 4.0.

Step 1 Validation: Testing with Same Simulation

26 Code Version. Human-Model agreement between Sonnet 4.6 codes assigned and the human codes assigned was 73.80% (versus 74.18% for Sonnet 4.0). Cohen's kappa (treating the model as one rater and the human coders as the second rater) was .70 (same for Sonnet 4.0). In both cases the no-information rate was .21 (*p* value of difference between Cohen's kappa and no-information rate was <.001). Estimated observer accuracy, using KappaAcc (Bakeman, 2022), was .85 (same as Sonnet 4.0). Overall, these results were nearly identical.

5 Code Version. Human-Model agreement between Sonnet 4.6 codes assigned and the human codes assigned was 77.48 (versus 77.81 for Sonnet 4.0). Cohen's kappa (treating the model as one rater and the human coders as the second rater) was .71 (same as Sonnet 4.0). In both cases the no-information rate was .26 (*p* value of difference between Cohen's kappa and no-information rate was <.001). Estimated observer accuracy, using KappaAcc (Bakeman, 2022), was 87% (same as Sonnet 4.0). Overall, these results were nearly identical.

Step 2 Validation: Testing with a Different Simulation

26 Code Version. Human-Model agreement between Sonnet 4.6 codes assigned and the human codes assigned was 53.91% for Sonnet 4.6 (versus 55.10% for Sonnet 4.0). Cohen's kappa (treating the model as one rater and the human coders as the second rater) was .51 (versus .49 for Sonnet 4.0). In both cases the no-information rate was .29 (*p* value of difference between Cohen's kappa and no-information rate was <.001). Estimated observer accuracy, using KappaAcc (Bakeman, 2022), was .74 (versus .72 for Sonnet 4.0). Overall, these results were nearly identical.

5 Code Version. Human-Model agreement between Sonnet 4.6 codes assigned and the human codes assigned was 62.82% for Sonnet 4.6 (versus 60.53% for Sonnet 4.0). Cohen's kappa (treating the model as one rater and the human coders as the second rater) was .52 (versus .49 for Sonnet 4.0). In both cases the no-information rate was .36 (*p* value of difference between Cohen's kappa and no-information rate was <.001). Estimated observer accuracy, using KappaAcc (Bakeman, 2022), was 78% (versus 76% for Sonnet 4.0). Overall, these results were nearly identical.

Together, these results indicate that Sonnet 4.6 performed much like Sonnet 4.0 in both validation steps.

[1] This 72% number from the KappaAcc program discussed above just happened to be the same as the 72% result we got for the mismatch analysis.

Formatting Your Transcripts

Set up your transcripts for analysis by putting them into an excel spreadsheet. The format should be as shown below. Label the first column "SpeakerName" and list whatever names you have for those speakers (e.g., buyer/seller, John/Mary). Label the second column "Content" and include the material that is contained in your unit of analysis (which may be a speaking turn, a sentence, or a thought unit). Also include columns for ResearcherName, Email, and Institution and include that information in the next row.

If you use *speaking turns* then speakers will alternate, and the format will look like this:

Speaker Name	Content	ResearcherName	Email	Institution
Buyer	*All words in speaking turn...*	*Your name*	*Your email*	*Your Institution*
Seller	*All words in speaking turn...*			
Buyer	*All words in speaking turn...*			
Seller	*All words in speaking turn...*			
Buyer	*All words in speaking turn...*			
etc	etc			

If you use *sentences* or *thought units* then it is possible that speakers may appear several times in a row, and the format will look like this:

Speaker Name	Content	ResearcherName	Email	Institution
Buyer	*All words in sentence or thought unit...*	*Your name*	*Your email*	*Your Institution*
Buyer	*All words in sentence or thought unit...*			
Buyer	*All words in sentence or thought unit...*			
Seller	*All words in sentence or thought unit...*			
Seller	*All words in sentence or thought unit...*			
Buyer	*All words in sentence or thought unit...*			
etc	etc			

Create one Excel file for each transcript. Name each file in the following way:

YourName_StudyName_1
YourName_StudyName_2
YourName_StudyName_3
Etc.

For example, my first file would be named "RayFriedman_CrownStudy_1" and the second file would be named "RayFriedman_CrownStudy_2", and so on.

Submit Your Transcript

To submit your transcript for the model to code, drag and drop one or several transcript files into the submission section of the website. It will likely take about 10 minutes for Claude to process each transcript, although this can vary based on how much demand Claude has at the moment you submit your files. Once the analysis for each transcript is complete, you can retrieve it on the download page using the codes shown when you submit your files. We suggest submitting just a few files at a time, so that you can check the output before doing too many analyses. The output files will include:

- Transcript Name
- Speaker
- The text (which could be a thought unit, sentence, or speaking turn)
- The code assigned to that text
- Consistency score for that code

References

Adair, W. L., & Brett, J. M. (2005). The negotiation dance: Time, culture, and behavioral sequences in negotiation. Organization Science, 16, 33-51.

Bakeman, R. (2022). KappaAcc: A program for assessing the adequacy of kappa. Behavior Research Methods. https://doi.org/10.3758/s13428-022-01836-1

Fleiss, J.L. (1981). Statistical methods for rates and proportions (2nd ed.). New York: John Wiley. ISBN 978-0-471-26370-8.

Friedman, R., Brett, J., Cho, J., Zhan, X. et al. (2024). An application large language models to coding negotiation transcripts: The Vanderbilt AI negotiation lab. (forthcoming)

Gamer, M., Lemon, J., Fellows, I., & Singh P. (2019) irr: Various coefficients of interrater reliability and agreement. R package version 0.84.1. https://CRAN.R-project.org/package=irr.

Garrison, D. Cleveland-Innes, M., Koole, M. & Kappelman, J. (2006). Revisiting methodological issues in transcript analysis: Negotiated coding and reliability. The Internet and Higher Education. 9. 1-8. 10.1016/j.iheduc.2005.11.001.

Gunia, B. C., Brett, J. M., Nandkeolyar, A. K., & Kamdar, D. (2011). Paying a price: Culture, trust, and negotiation consequences. Journal of Applied Psychology, 96, 774-789.

Jackel, E., Zerres, A., Hamshorn de Sanchez, C., Lehmann-Willenbrock, & N., Huffmeier, J. (2022), "NegotiAct: Introducing a comprehensive coding scheme to capture temporal interaction patterns in negotiations," *Group and Organization Management*. (See supplementary file for coding guidelines.)

Landis, J.R. & Koch, G.G. (1977). The measurement of observer agreement for categorical data. *Biometrics*. 33 (1): 159–174. doi:10.2307/2529310.

Kern, M. C., Brett, J. M., Weingart, L. R., & Eck, C. S. (2020). The "fixed" pie perception and strategy in dyadic versus multiparty negotiations. *Organizational Behavior and Human Decision Processes, 157,* 143–158. https://doi.org/10.1016/j.obhdp.2020.01.001

R Core Team (2022). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

Weingart, L. R., Thompson, L. L., Bazerman, M. H., & Carroll, J. S. (1990). Tactical behavior and negotiation outcomes. International Journal of Conflict Management, 1, 7-31

Xie. S.M. & Min, S. (2022). How does in-context learning work? A framework for understanding the differences from traditional supervised learning. Stanford AI Lab Blog, Aug 1. (https://ai.stanford.edu/blog/understanding-incontext/)

Appendix 1

Mapping of Kern et al.'s (2020) 37 Codes into Four Strategic Clusters

A section of Table 1 from Kern et al. (2020), p.149.

Strategic clusters and coded negotiation behaviors.

Strategic orientation	Strategic function	Strategy	Behavior included in strategy cluster
Integrative	Information expression and exploration	Integrative information	Acknowledgement without agreement Agreement with offer Agreement with statement Process suggestion to address one issue at a time Shows insight (summarizes others' interests) Notes task similarities Notes similarities in preferences and priorities Notes mutual interests Suggest moving on without resolution
	Action	Value creating	Ask questions about offer Provide information about issue priorities Make multi-issue offer Suggest package trade-off Suggest compromise Ask for priorities Suggest reciprocity – concession now in exchange for future concession
Distributive	Information expression and exploration	Distributive information	States issue preferences Asks for preferences Provide information about bottom line Ask for bottom line Makes statements about facts or task clarification Note differences in preferences and priorities Note task differences
	Action	Value claiming	Substantiation of position Make single-issue offer Disagree with statement Ask about others' substantiation Disagree with offer made Refer to power Make threats

Note that this Table only includes 30 of the 37 behaviors. As Kern et a.l (2020) report in the footnote to this table, six behaviors were omitted "because they were procedural or miscellaneous, did not fit into one of the four main clusters, or because they had zero frequency in either the dyadic or the multiparty sample." The codes not included in their four strategic clusters were:

1. Ask miscellaneous task related questions
2. Vote on one issue – before moving on to the next
3. Other procedural comments – procedure for managing the discussion
4. Time – time checks
5. Creative solutions – potential solutions outside boundaries of task
6. Miscellaneous - general on task-related statements and comments
7. Off-task questions, answers, and/or comments

Appendix 2: Conversion of Original 37 Codes to 26 Codes (also available here)

Code	Code Description	Strategic Categooory	Combination	New Label	Combined Code Description
OFFER					
OS	SINGLE-ISSUE – secure agreement on one issue	Distributive Action	n/a	n/a	
OM	MULTI-ISSUE - secure agreement on two or more issues	Integrative Action	n/a	n/a	
PROVIDE INFORMATION					
IP	ISSUE PREFERENCES - Within a single issue	Distrib Info	n/a	n/a	
IB	BOTTOM-LINE – within a single issue or for a package	Distrib Info	n/a	n/a	
IR	PRIORITIES - Relative importance of issue(s)	Integrative Action	n/a	n/a	
SUBSTANTIATION					
SB	DEFENDING ARGUMENTS – argue position on issue	Distributive Action	n/a	n/a	
SF	FACTUAL STATEMENTS – facts or task clarifications that are specific/true	Distrib Info	n/a	n/a	
QUESTIONS					
QO	ASK ABOUT OFFER – Clarification or relating to an offer*	Integrative Action	n/a	n/a	
QR	ASK FOR PRIORITIES - Relative importance of issue(s)	Integrative Action	n/a	n/a	
QP	ASK FOR PREFERENCES - Within a single issue	Distrib Info	n/a	n/a	
QB	ASK FOR BOTTOM LINE –within a single issue or package	Distrib Info	n/a	n/a	
QS	ASK ABOUT SUBSTANTIATION – question/clarification of argument presented	Distributive Action	n/a	n/a	
QM	ASK MISCELLANEOUS TASK RELATED QUESTIONS	EXCLUDED from strategic categories by Kern et al.	n/a	n/a	
SUMMARIZING					
IN	INSIGHT - summarizing others' interests	Integrtive Info	n/a	n/a	
MU	MUTUALITY - noting mutual interests	Integrtive Info	n/a	n/a	
GS	GENERAL SIMILARITIES – which are task related	Integrtive Info	IS+GS	Similarities - SIM	Noting similaties related to preference, priorities, or other aspects of the negotiation
IS	SIMILARITIES – noting similarities in issue related preferences and priorities	Integrtive Info	IS+GS	Similarities - SIM	
ID	DIFFERENCES – noting differences in issue related preferences and priorities	Distrib Info	ID+GD	Differences -DIFF	Noting differences related to preference, priorities, or other aspects of the negotiation
GD	GENERAL DIFFERENCES – which are task related	Distrib Info	ID+GD	Differences -DIFF	
THREATS/POWER					
TH	THREAT – action if others do not comply	Distributive Action	TH+PW	Coercive - COER	Actions designed to threaten the other party or show one's ability to dominate over the
PW	POWER – one's ability to dominate others	Distributive Action	TH+PW	Coercive - COER	
REACTIONS					
RPO	AGREEMENT TO OFFER MADE	Integrative Info	RPO+RPS+RPSA	Agreement - AGR	Express agreement with an offer or statement, or simple acknowledging what the other party says
RPS (RPSA)	AGREEMENT WITH STATEMENT – agreement with what was said	Integrative Info	RPO+RPS+RPSA	Agreement - AGR	
RPA (RPSA)	ACKNOWLEDGEMENT WITHOUT AGREEMENT – Also includes back channeling	Integrative Info	RPO+RPS+RPSA	Agreement - AGR	
RNO	DISAGREEMENT WITH OFFER MADE	Distributive Action	RNO+RNS	Disagreement - DIS	Expressing disagreement with an offer or statement
RNS	DISAGREEMENT WITH STATEMENT – disagree with what was said	Distributive Action	RNO+RNS	Disagreement - DIS	
PROCEDURAL COMMENTS					
PC	COMPROMISE - suggest compromise or willingness to concede	Integrative Action	PC+PP+PX	Integrative Procedure - INTPROC	Offering a compromise or trade-off or exchange with the other party
PP	PACKAGE/TRADE-OFF – involving two or more issues	Integrative Action	PC+PP+PX	Integrative Procedure - INTPROC	
PX	RECIPROCITY – concession in exchange for future concession	Integrative Action	PC+PP+PX	Integrative Procedure - INTPROC	
PI	DISCUSS ONE ISSUE - deal with one issue at a time	Integrative Info	n/a	n/a	
PM	MOVING ON - without resolution	Integrtive Info	n/a	n/a	
PV	VOTE ON ONE ISSUE – before moving on to the next	EXCLUDED from strategic categories by Kern et al.	n/a	n/a	Excluded from revised coding since not present in the data
PT	TIME – time checks	EXCLUDED from strategic categories by Kern et al.	n/a	n/a	
PO	OTHER – procedure for managing the discussion	EXCLUDED from strategic categories by Kern et al.	PO+OT+MI	Miscelaneous - MISC	Miscelaneous comments about procedures. Does not fit into other coding categories.
MISCELLANEOUS /OTHER					
MI	MISCELLANEOUS - general on-task related statements and comments	EXCLUDED from strategic categories by Kern et al.	PO+OT+MI	Miscelaneous - MISC	
OT	OFF-TASK QUESTIONS, ANSWERS, AND/OR COMMENTS	EXCLUDED from strategic categories by Kern et al.	PO+OT+MI	Miscelaneous - MISC	
CS	CREATIVE SOLUTIONS – potential solutions outside boundaries of task	EXCLUDED from strategic categories by Kern et al.	n/a	n/a	